Moderation

Introduction to SEM with Lavaan

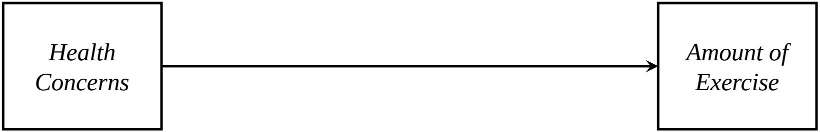
Kyle M. Lang

Department of Methodology & Statistics
Utrecht University

Outline

Refresher: Focal Effect Only

The *healthConcerns* → *exerciseAmount* relation is our *focal effect*

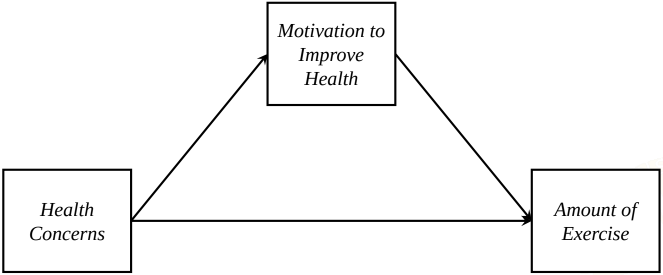


- Mediation, moderation, and conditional process analysis all attempt to describe the focal effect in more detail.
- We always begin by hypothesizing a focal effect.

Refresher: Mediation Hypothesis

A mediation analysis will attempt to describe how health concerns affect amount of exercise.

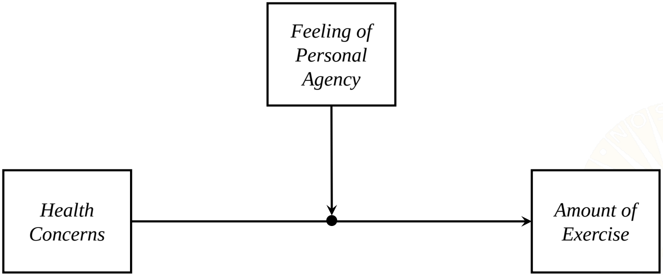
- The *how* is operationalized in terms of intermediary variables.
- Mediator: Motivation to improve health (*motivation*).



Refresher: Moderation Hypothesis

A moderation hypothesis will attempt to describe when health concerns affect amount of exercise.

- The *when* is operationalized in terms of interactions between the focal predictor and contextualizing variables
- Moderator: Sense of personal agency relating to physical health (*agency*).



Equations

In additive MLR, we might have the following equation:

$$Y = \beta_0 + \beta_1 X + \beta_2 Z + \varepsilon$$

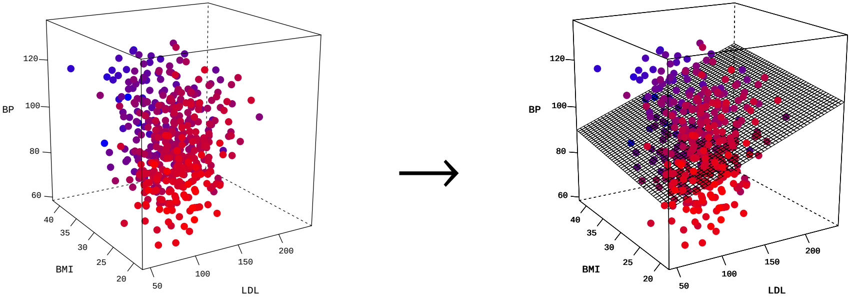
This additive equation assumes that X and Z are independent predictors of Y.

When X and Z are independent predictors, the following are true:

- X and Z *can* be correlated.
- β_1 and β_2 are *partial* regression coefficients.
- The effect of X on Y is the same at **all levels** of Z, and the effect of Z on Y is the same at **all levels** of X.

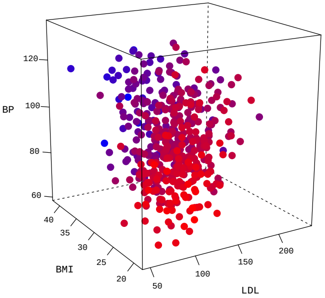
Additive Regression

The effect of X on Y is the same at **all levels** of Z.

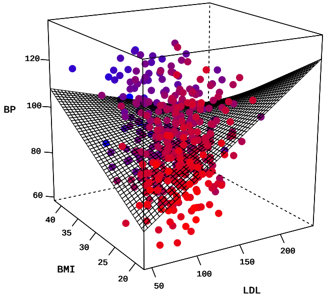


Moderated Regression

The effect of X on Y varies **as a function** of Z.



$\longrightarrow$



Equations

The following derivation is adapted from **?**.

- When testing moderation, we hypothesize that the effect of X on Y varies as a function of Z.
- We can represent this concept with the following equation:

$$Y = \beta_0 + f(Z)X + \beta_2 Z + \varepsilon \tag{1}$$

Equations

The following derivation is adapted from **?**.

- When testing moderation, we hypothesize that the effect of X on Y varies as a function of Z.
- We can represent this concept with the following equation:

$$Y = \beta_0 + f(Z)X + \beta_2 Z + \varepsilon \tag{1}$$

- If we assume that Z linearly (and deterministically) affects the relationship between X and Y, then we can take:

$$f(Z) = \beta_1 + \beta_3 Z \tag{2}$$

Equations

- Substituting Equation 2 into Equation 1 leads to:

$$Y = \beta_0 + (\beta_1 + \beta_3 Z)X + \beta_2 Z + \varepsilon$$

Equations

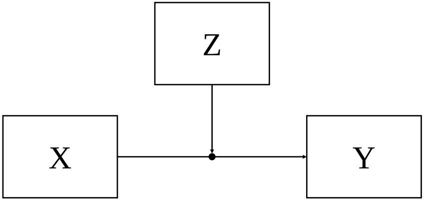
- Substituting Equation 2 into Equation 1 leads to:

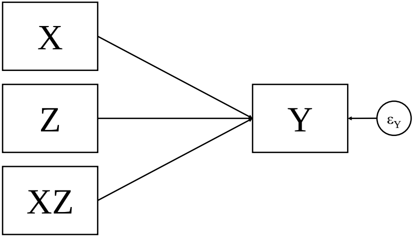
$$Y = \beta_0 + (\beta_1 + \beta_3 Z)X + \beta_2 Z + \varepsilon$$

- Which, after distributing X and reordering terms, becomes:

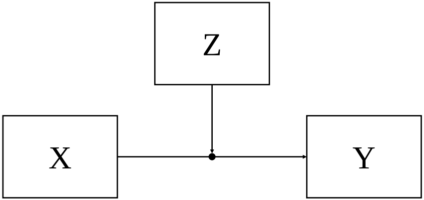
$$Y = \beta_0 + \beta_1 X + \beta_2 Z + \beta_3 XZ + \varepsilon$$

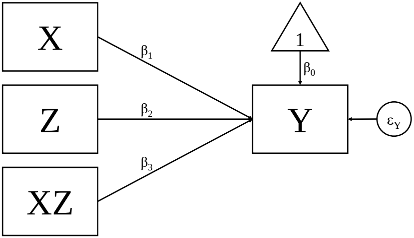
Conceptual vs. Analytic Diagrams





Conceptual vs. Analytic Diagrams





Testing Moderation

Now, we have an estimable regression model that quantifies the linear moderation we hypothesized.

$$\boxed{Y = \beta_0 + \beta_1 X + \beta_2 Z + \beta_3 XZ + \varepsilon}$$

- To test for significant moderation, we simply need to test the significance of the interaction term, XZ.
  - Check if $\hat{\beta}_3$ is significantly different from zero.

Interpretation

Given the following equation:

$$Y = \hat{\beta}_0 + \hat{\beta}_1 X + \hat{\beta}_2 Z + \hat{\beta}_3 XZ + \hat{\varepsilon}$$

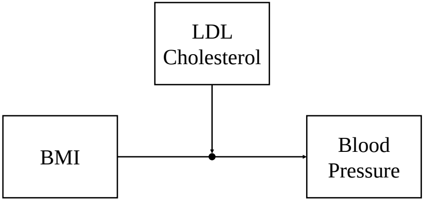
- $\hat{\beta}_3$ quantifies the effect of Z on the focal effect (the $X \rightarrow Y$ effect).
  - For a unit change in Z, $\hat{\beta}_3$ is the expected change in the effect of X on Y.
- $\hat{\beta}_1$ and $\hat{\beta}_2$ are *conditional effects*.
  - Interpreted where the other predictor is zero.
  - For a unit change in X, $\hat{\beta}_1$ is the expected change in Y, when $Z = 0$.
  - For a unit change in Z, $\hat{\beta}_2$ is the expected change in Y, when $X = 0$.

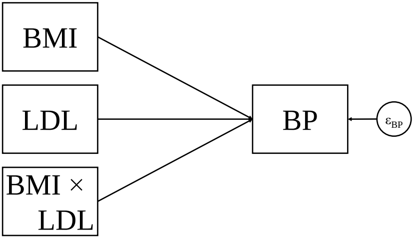
Example

Looking at the *diabetes* dataset.

- We suspect that patients' BMIs are predictive of their average blood pressure.
- We further suspect that this effect may be differentially expressed depending on the patients' LDL levels.

Diagrams


LDL Cholesterol
BMI
Blood Pressure


BMI
LDL
BMI × LDL
BP
ε_{BP}

Example

```r
dDat <- readRDS("../data/diabetes.rds")

## Focal Effect:
out0 <- lm(bp ~ bmi, data = dDat)
partSummary(out0, -c(1, 2))
```

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  61.9973     3.6659   16.91   <2e-16
bmi           1.2379     0.1371    9.03   <2e-16

Residual standard error: 12.72 on 440 degrees of freedom
Multiple R-squared:  0.1563,Adjusted R-squared:  0.1544
F-statistic: 81.54 on 1 and 440 DF,  p-value: < 2.2e-16
```

Example

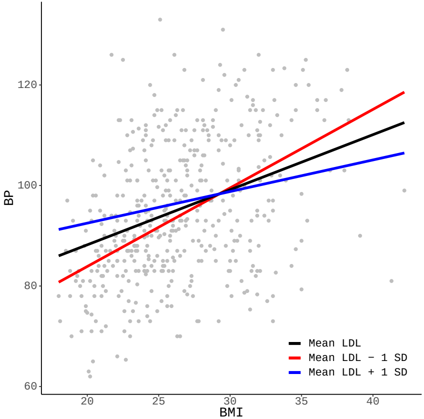
```
## Additive Model:
out1 <- lm(bp ~ bmi + ldl, data = dDat)
partSummary(out1, -c(1, 2))

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 59.26577    3.91281  15.147  < 2e-16
bmi          1.16567    0.14156   8.235 2.08e-15
ldl          0.04016    0.02056   1.953   0.0515

Residual standard error: 12.68 on 439 degrees of freedom
Multiple R-squared:  0.1636,Adjusted R-squared:  0.1598
F-statistic: 42.94 on 2 and 439 DF,  p-value: < 2.2e-16
```

Example

```
## Moderated Model:
out2 <- lm(bp ~ bmi * ldl, data = dDat)
partSummary(out2, -c(1, 2))

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 14.480616  14.291677   1.013 0.311514
bmi          2.867825   0.541312   5.298 1.86e-07
ldl          0.448771   0.127160   3.529 0.000461
bmi:ldl     -0.015352   0.004716  -3.255 0.001221

Residual standard error: 12.54 on 438 degrees of freedom
Multiple R-squared:  0.1834,Adjusted R-squared:  0.1778
F-statistic: 32.78 on 3 and 438 DF,  p-value: < 2.2e-16
```

Visualizing the Interaction

We can get a better idea of the patterns of moderation by plotting the focal effect at conditional values of the moderator.



Example

Of course, we can fit the same model in **lavaan**.

```
library(lavaan)

## Specify the model:
mod <- "bp ~ 1 + bmi + ldl + bmi:ldl"

## Use a dplyr pipeline to create the product term and estimate the model:
lavOut <- sem(mod1, data = dDat)

Error in lavaan::lavaan(model = mod1, data = dDat, model.type = "sem", :
object 'mod1' not found
```

Example

```
partSummary(lavOut, 5:7)

Error in h(simpleError(msg, call)):  error in evaluating the argument
'object' in selecting a method for function 'summary':  object 'lavOut' not
found
```

POST HOC ANALYSIS

Probing the Interaction

A significant estimate of β_3 tells us that the effect of X on Y depends on the level of Z, but nothing more.

- The plot above gives a descriptive illustration of the pattern, but does not support statistical inference.
  - The three conditional effects we plotted look different, but we cannot say much about how they differ with only the plot and $\hat{\beta}_3$.
- This is the purpose of *probing* the interaction.
  - Try to isolate areas of Z's distribution in which $X \rightarrow Y$ effect is significant and areas where it is not.

Probing the Interaction

The most popular method of probing interactions is to do a so-called *simple slopes* analysis.

- Pick-a-point approach
- Spotlight analysis

In simple slopes analysis, we test if the slopes of the conditional effects plotted above are significantly different from zero.

- To do so, we test the significance of *simple slopes*.

Simple Slopes

Recall the derivation of our moderated equation:

$$Y = \beta_0 + \beta_1 X + \beta_2 Z + \beta_3 XZ + \varepsilon$$

We can reverse the process by factoring out X and reordering terms:

$$Y = \beta_0 + (\beta_1 + \beta_3 Z)X + \beta_2 Z + \varepsilon$$

Where $f(Z) = \beta_1 + \beta_3 Z$ is the linear function that shows how the relationship between X and Y changes as a function of Z.

$f(Z)$ is the *simple slope*.

- By plugging different values of Z into $f(Z)$, we get the value of the conditional effect of X on Y at the chosen level of Z.

Significance Testing of Simple Slopes

The values of Z used to define the simple slopes are arbitrary.

- The most common choice is: $\{(\bar{Z} - SD_Z), \bar{Z}, (\bar{Z} + SD_Z)\}$
- You could also use interesting percentiles of Z's distribution.

The standard error of a simple slope is given by:

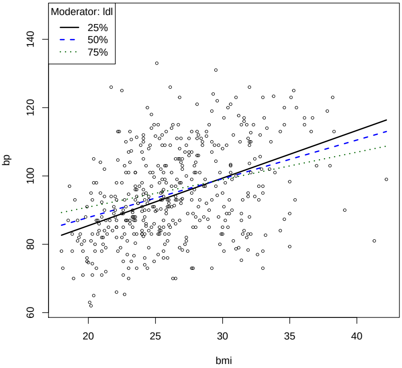
$$SE_{f(Z)} = \sqrt{SE^2_{\beta_1} + 2Z \cdot \text{COV}(\beta_1, \beta_3) + Z^2 SE^2_{\beta_3}}$$

So, you can test the significance of a simple slope by constructing a Wald statistic or confidence interval using $\hat{f}(Z)$ and $SE_{f(Z)}$:

$$t = \frac{\hat{f}(Z)}{SE_{f(Z)}}, \quad CI = \hat{f}(Z) \pm t_{crit} \times SE_{f(Z)}$$

Example

```r
library(rockchalk)

## Prepare the metadata with rockchalk::plotSlopes():
psOut <- plotSlopes(out2, plotx = "bmi", modx = "ldl")
```



Example

```
## Test the simple slopes with rockchalk::testSlopes():
tsOut <- testSlopes(psOut)

Values of ldl OUTSIDE this interval:
      lo        hi
154.7983 305.9235
cause the slope of (b1 + b2*ldl)bmi to be statistically significant

tsOut$hypotests

     "ldl"     slope Std. Error  t value     Pr(>|t|)
25%  96.05 1.3932882  0.1565225 8.901520 1.471778e-17
50% 113.00 1.1330757  0.1403937 8.070700 6.794806e-15
75% 134.50 0.8030128  0.1789442 4.487505 9.222263e-06
```

Example

We can use **semTools** routines to probe interaction in **lavaan** models.

- `probe2WayMC()`: simple slopes/intercepts analysis
- `plotProbe()`: simple slopes plots

```
library(semTools)

## Estimate and test simple slopes and simple intercepts:
ssOut <- probe2WayMC(lavOut,
                     nameX = c("bmi", "ldl", "bmi:ldl"),
                     nameY = "bp",
                     modVar = "ldl",
                     valProbe = quantile(dDat$ldl, c(0.25, 0.50, 0.75))
                     )

Error in probe2WayMC(lavOut, nameX = c("bmi", "ldl", "bmi:ldl"), nameY =
"bp", :  object 'lavOut' not found
```

Example

```
## View the results:
ssOut

Error in eval(expr, envir, enclos):  object 'ssOut' not found
```

Example

```
## Plot the simple slopes:
plotProbe(ssOut, xlim = range(dDat$bmi), xlab = "BMI", ylab = "BP")

Error in plotProbe(ssOut, xlim = range(dDat$bmi), xlab = "BMI", ylab =
"BP"):  object 'ssOut' not found
```

Latent Variable Interactions

Latent Variable Interactions

When we have two observed variables interacting to predict a latent variable, our job is easy:

1. Construct the product term of the observed focal and moderator variables
2. Use the observed focal, moderator, and interaction variables to predict the latent DV

If we want to model moderation when at least on of the predictors is latent, things get more difficult.

- If the moderator is observed and discrete, we can use multiple group modeling
- If the moderator is continuous and/or latent, then we need fancier methods

Two basic approaches:

1. Methods based on products of manifest variables
2. Methods based on directly estimating the products of latent variables

Computing Interaction Indicators

The simplest approach is to create observed product terms and directly use those terms as indicators of the interaction construct.

- Naively indicating an interaction construct with the raw product terms is probably sub-optimal
- Collinearity among the interaction indicators and the raw items can cause estimation problems
- From a modeling perspective, we'd like to interpret out final model holistically

Two recommended approaches:

1. Orthogonalization through residual centering (Little, Bovaird, & Widaman, 2006).
2. Double mean centering (Lin, Wen, Marsh, & Lin, 2010).

Orthogonalization

Say we want to estimate the moderated effect of Z on the $X \rightarrow Y$ effect, where X, Y, and Z are latent variables indicated by $\{x_1, x_2, x_3\}$, $\{y_1, y_2, y_3\}$, and $\{z_1, z_2, z_3\}$, respectively.

Orthogonalization is performed by:

1. Construct all possible product terms: $\{x_1z_1, x_1z_2, x_1z_3, x_2z_1, x_2z_2, x_2z_3, x_3z_1, x_3z_2, x_3z_3\}$.
2. Regress each product term onto all observed indicators of X and Z:

$$\begin{aligned}
\widehat{x_1z_1} &= \alpha + \beta_1 x_1 + \beta_2 x_2 + \beta_2 x_3 + \beta_4 z_1 + \beta_5 z_2 + \beta_6 z_3 \\
\widehat{x_2z_1} &= \alpha + \beta_1 x_1 + \beta_2 x_2 + \beta_2 x_3 + \beta_4 z_1 + \beta_5 z_2 + \beta_6 z_3 \\
&\vdots \\
\widehat{x_3z_3} &= \alpha + \beta_1 x_1 + \beta_2 x_2 + \beta_2 x_3 + \beta_4 z_1 + \beta_5 z_2 + \beta_6 z_3
\end{aligned}$$

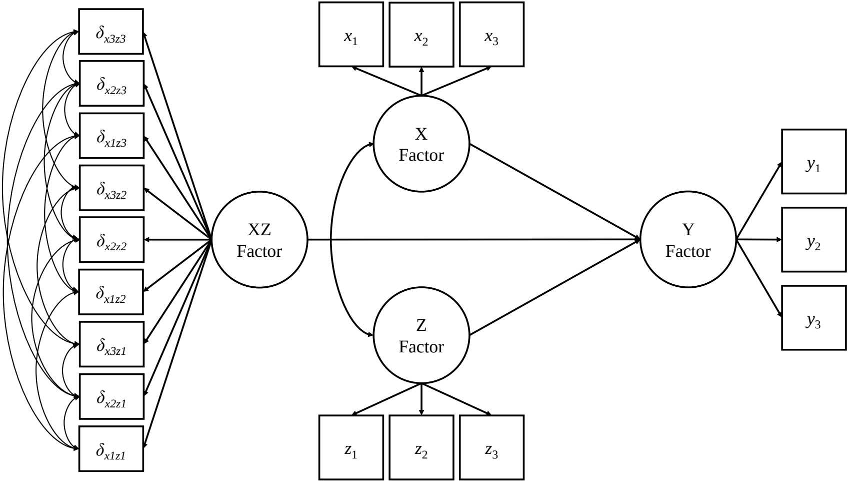
Orthogonalization

3. Calculate each product term's residual:

$$
\begin{aligned}
\delta_{x1z1} &= x_1 z_1 - \widehat{x_1 z_1} \\
\delta_{x1z1} &= x_2 z_1 - \widehat{x_2 z_1} \\
&\vdots \\
\delta_{x3z3} &= x_3 z_3 - \widehat{x_3 z_3}
\end{aligned}
$$

4. Use these residuals to indicate a latent interaction construct.

Orthogonalization



Example

```r
dat1 <- readRDS("../data/lecture12Data.rds")

mod1 <- "
fX =~ x1 + x2 + x3
fZ =~ z1 + z2 + z3
fY =~ y1 + y2 + y3
"

out1 <- cfa(mod1, data = dat1, std.lv = TRUE)
```

Example

```
partSummary(out1, 1:4)

lavaan 0.6-11 ended normally after 17 iterations

  Estimator                                         ML
  Optimization method                           NLMINB
  Number of model parameters                        21

  Number of observations                           500

Model Test User Model:

  Test statistic                                41.021
  Degrees of freedom                                24
  P-value (Chi-square)                           0.017

Parameter Estimates:

  Standard errors                             Standard
  Information                                 Expected
  Information saturated (h1) model          Structured
```

Example

```
partSummary(out1, 5)

Latent Variables:
                   Estimate  Std.Err  z-value  P(>|z|)
  fX =~
    x1                0.671    0.044   15.407    0.000
    x2                0.661    0.043   15.226    0.000
    x3                0.702    0.045   15.481    0.000
  fZ =~
    z1                0.738    0.048   15.343    0.000
    z2                0.734    0.048   15.157    0.000
    z3                0.718    0.046   15.601    0.000
  fY =~
    y1                0.787    0.045   17.614    0.000
    y2                0.729    0.045   16.325    0.000
    y3                0.761    0.043   17.797    0.000
```

Example

```
partSummary(out1, 6)

Covariances:
                   Estimate  Std.Err  z-value  P(>|z|)
  fX ~~
    fZ                0.232    0.058    3.987    0.000
    fY                0.827    0.033   25.310    0.000
  fZ ~~
    fY                0.156    0.057    2.739    0.006
```

Example

```
partSummary(out1, 7)

Variances:
                   Estimate  Std.Err  z-value  P(>|z|)
   .x1                0.510    0.042   11.998    0.000
   .x2                0.514    0.042   12.141    0.000
   .x3                0.550    0.046   11.938    0.000
   .z1                0.523    0.052   10.141    0.000
   .z2                0.546    0.052   10.443    0.000
   .z3                0.461    0.048    9.706    0.000
   .y1                0.492    0.044   11.185    0.000
   .y2                0.545    0.044   12.253    0.000
   .y3                0.444    0.040   11.007    0.000
    fX                1.000
    fZ                1.000
    fY                1.000
```

Example

```
fitMeasures(out1, c("chisq", "df", "pvalue", "cfi", "tli", "rmsea", "srmr"))

 chisq      df  pvalue    cfi    tli   rmsea   srmr
41.021 24.000   0.017  0.987  0.981  0.038  0.026
```

Example

Now, we'll fit the additive model as an SEM.

```
mod2 <- "
fX =~ x1 + x2 + x3
fZ =~ z1 + z2 + z3
fY =~ y1 + y2 + y3

fY ~ fX + fZ
"

out2 <- sem(mod2, data = dat1, std.lv = TRUE)
```

Example

```
partSummary(out2, 1:4)

lavaan 0.6-11 ended normally after 22 iterations

  Estimator                                         ML
  Optimization method                           NLMINB
  Number of model parameters                        21

  Number of observations                           500

Model Test User Model:

  Test statistic                                41.021
  Degrees of freedom                                24
  P-value (Chi-square)                           0.017

Parameter Estimates:

  Standard errors                             Standard
  Information                                 Expected
  Information saturated (h1) model          Structured
```

Example

```
partSummary(out2, 5)

Latent Variables:
                   Estimate  Std.Err  z-value  P(>|z|)
  fX =~
    x1                0.671    0.044   15.407    0.000
    x2                0.661    0.043   15.226    0.000
    x3                0.702    0.045   15.481    0.000
  fZ =~
    z1                0.738    0.048   15.343    0.000
    z2                0.734    0.048   15.157    0.000
    z3                0.718    0.046   15.601    0.000
  fY =~
    y1                0.442    0.044   10.079    0.000
    y2                0.409    0.041    9.877    0.000
    y3                0.427    0.042   10.099    0.000
```

Example

```
partSummary(out2, 6:7)

Regressions:
                   Estimate  Std.Err  z-value  P(>|z|)
  fY ~
    fX                1.488    0.190    7.820    0.000
    fZ               -0.066    0.090   -0.732    0.464

Covariances:
                   Estimate  Std.Err  z-value  P(>|z|)
  fX ~~
    fZ                0.232    0.058    3.987    0.000
```

Example

```
partSummary(out2, 8)

Variances:
                   Estimate  Std.Err  z-value  P(>|z|)
   .x1                0.510    0.042   11.998    0.000
   .x2                0.514    0.042   12.141    0.000
   .x3                0.550    0.046   11.938    0.000
   .z1                0.523    0.052   10.141    0.000
   .z2                0.546    0.052   10.443    0.000
   .z3                0.461    0.048    9.706    0.000
   .y1                0.492    0.044   11.185    0.000
   .y2                0.545    0.044   12.253    0.000
   .y3                0.444    0.040   11.007    0.000
    fX                1.000
    fZ                1.000
   .fY                1.000
```

Example

```r
fitMeasures(out2, c("chisq", "df", "pvalue", "cfi", "tli", "rmsea", "srmr"))

 chisq      df  pvalue     cfi     tli   rmsea    srmr
41.021  24.000   0.017   0.987   0.981   0.038   0.026
```

Example

```r
library(dplyr)

## Set aside a copy of the predictor data for later use:
preds <- select(dat1, matches("x\\d|z\\d")) %>% as.matrix()

## Construct product terms:
products <- mutate(dat1,
                   x1z1 = x1 * z1,
                   x1z2 = x1 * z2,
                   x1z3 = x1 * z3,

                   x2z1 = x2 * z1,
                   x2z2 = x2 * z2,
                   x2z3 = x2 * z3,

                   x3z1 = x3 * z1,
                   x3z2 = x3 * z2,
                   x3z3 = x3 * z3,
                   .keep = "none")
```

Example

```r
## Residualize the product terms:
products <- sapply(products,
                   function(y, x) lm(y ~ x)$resid,
                   x = preds)

## Join data pieces:
dat2 <- data.frame(dat1, products)
```

Example

```
mod3 <- "
fX  =~ x1 + x2 + x3
fZ  =~ z1 + z2 + z3
fY  =~ y1 + y2 + y3
fXZ =~ x1z1 + x1z2 + x1z3 + x2z1 + x2z2 + x2z3 + x3z1 + x3z2 + x3z3

fY ~ fX + fZ + fXZ

fX ~~ 0*fXZ
fZ ~~ 0*fXZ

x1z1 ~~ x1z2 + x1z3 + x2z1 + x3z1
x1z2 ~~ x1z3 + x2z2 + x3z2
x1z3 ~~ x2z3 + x3z3

x2z1 ~~ x2z2 + x2z3 + x3z1
x2z2 ~~ x2z3 + x3z2
x2z3 ~~ x3z3

x3z1 ~~ x3z2 + x3z3
x3z2 ~~ x3z3
"
```

Example

```
out3 <- sem(mod3, data = dat2, std.lv = TRUE, meanstructure = TRUE)

partSummary(out3, 5, 1:14)

Latent Variables:
                   Estimate  Std.Err  z-value  P(>|z|)
  fX =~
    x1                0.670    0.043   15.424    0.000
    x2                0.660    0.043   15.256    0.000
    x3                0.704    0.045   15.569    0.000
  fZ =~
    z1                0.738    0.048   15.342    0.000
    z2                0.734    0.048   15.156    0.000
    z3                0.718    0.046   15.602    0.000
  fY =~
    y1                0.396    0.046    8.545    0.000
    y2                0.369    0.044    8.441    0.000
    y3                0.383    0.045    8.558    0.000
```

Example

```
partSummary(out3, 5, c(1, 2, 15:24))

Latent Variables:
                   Estimate  Std.Err  z-value  P(>|z|)
  fXZ =~
    x1z1              0.361    0.053    6.833    0.000
    x1z2              0.427    0.056    7.615    0.000
    x1z3              0.432    0.053    8.190    0.000
    x2z1              0.558    0.056    9.914    0.000
    x2z2              0.616    0.062   10.008    0.000
    x2z3              0.520    0.057    9.153    0.000
    x3z1              0.516    0.059    8.805    0.000
    x3z2              0.626    0.063   10.007    0.000
    x3z3              0.521    0.058    8.936    0.000
```

Example

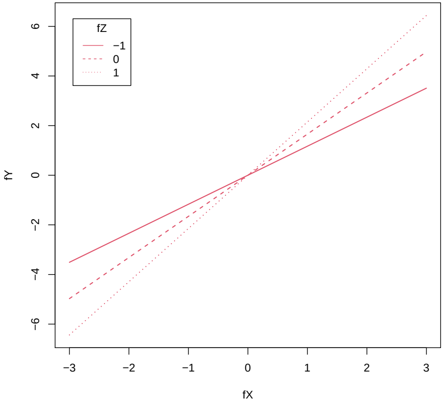
```
partSummary(out3, 6:7, -(14:39))

Regressions:
                   Estimate  Std.Err  z-value  P(>|z|)
  fY ~
    fX                1.658    0.239    6.930    0.000
    fZ               -0.074    0.099   -0.750    0.453
    fXZ               0.488    0.120    4.049    0.000

Covariances:
                   Estimate  Std.Err  z-value  P(>|z|)
  fX ~~
    fXZ               0.000
  fZ ~~
    fXZ               0.000
  fX ~~
    fZ                0.232    0.058    3.987    0.000
```

Example

```
fitMeasures(out3, c("chisq", "df", "pvalue", "cfi", "tli", "rmsea", "srmr"))

  chisq      df  pvalue     cfi     tli   rmsea    srmr
 74.899 113.000   0.998   1.000   1.015   0.000   0.019

probeOut3 <- probe2WayRC(fit      = out3,
                         nameX    = c("fX", "fZ", "fXZ"),
                         nameY    = "fY",
                         modVar   = "fZ",
                         valProbe = c(-1, 0, 1)
                         )

probeOut3$SimpleSlope

  fZ   est    se     z pvalue
1 -1 1.170 0.207 5.653      0
2  0 1.658 0.238 6.974      0
3  1 2.145 0.310 6.923      0
```

Example

```
plotProbe(probeOut3, xlim = c(-3, 3), xlab = "fX", ylab = "fY")
```



Double Mean Centering

We could also specify the interaction factor using double mean centering.

1. Mean center every indicator of X and Z:

$$x_1^c = x_1 - \bar{x}_1$$
$$\vdots$$
$$z_1^c = z_1 - \bar{z}_1$$
$$\vdots$$

2. Use the centered indicators to construct all possible product terms: $\{x_1^c z_1^c, x_1^c z_2^c, x_1^c z_3^c, x_2^c z_1^c, x_2^c z_2^c, x_2^c z_3^c, x_3^c z_1^c, x_3^c z_2^c, x_3^c z_3^c\}$.

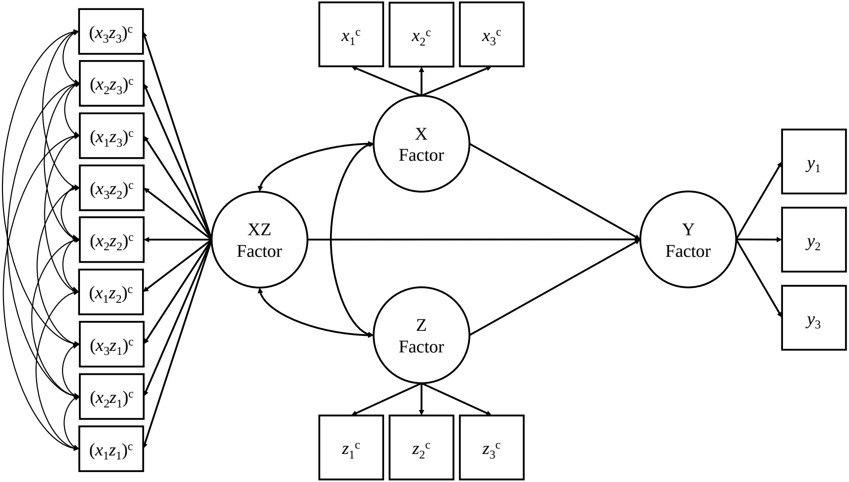
Double Mean Centering

3. Mean center each product term:

$$\begin{gathered}(x_1 z_1)^c = x_1^c z_1^c - \overline{x_1^c z_1^c} \\ (x_1 z_2)^c = x_1^c z_2^c - \overline{x_1^c z_2^c} \\ \vdots \\ (x_3 z_3)^c = x_3^c z_3^c - \overline{x_3^c z_3^c}\end{gathered}$$

4. Use the mean centered indicators of X and Z, and the “double mean centered” product terms to specify the latent interaction model.

Double Mean Centering



Example

```r
## Mean-center the predictor variables:
preds <- scale(preds, scale = FALSE) %>% as.data.frame()

## Construct and mean-center the product terms:
products <- mutate(preds,
                   x1z1 = x1 * z1,
                   x1z2 = x1 * z2,
                   x1z3 = x1 * z3,

                   x2z1 = x2 * z1,
                   x2z2 = x2 * z2,
                   x2z3 = x2 * z3,

                   x3z1 = x3 * z1,
                   x3z2 = x3 * z2,
                   x3z3 = x3 * z3,
                   .keep = "none") %>%
    scale(scale = FALSE)

## Join the data pieces:
dat3 <- select(dat1, matches("y\\d")) %>% data.frame(preds, products)
```

Example

```
mod4 <- "
fX =~ x1 + x2 + x3
fZ =~ z1 + z2 + z3
fY =~ y1 + y2 + y3
fXZ =~ x1z1 + x1z2 + x1z3 + x2z1 + x2z2 + x2z3 + x3z1 + x3z2 + x3z3

fY ~ fX + fZ + fXZ

x1z1 ~~ x1z2 + x1z3 + x2z1 + x3z1
x1z2 ~~ x1z3 + x2z2 + x3z2
x1z3 ~~ x2z3 + x3z3

x2z1 ~~ x2z2 + x2z3 + x3z1
x2z2 ~~ x2z3 + x3z2
x2z3 ~~ x3z3

x3z1 ~~ x3z2 + x3z3
x3z2 ~~ x3z3
"
```

Example

```
## Estimate the model:
out4 <- sem(mod4, data = dat3, std.lv = TRUE)

partSummary(out4, 5, 1:14)

Latent Variables:
                   Estimate  Std.Err  z-value  P(>|z|)
  fX =~
    x1                0.673    0.043   15.555    0.000
    x2                0.659    0.043   15.260    0.000
    x3                0.702    0.045   15.569    0.000
  fZ =~
    z1                0.738    0.048   15.360    0.000
    z2                0.734    0.048   15.154    0.000
    z3                0.718    0.046   15.597    0.000
  fY =~
    y1                0.386    0.048    8.009    0.000
    y2                0.359    0.045    7.925    0.000
    y3                0.373    0.047    8.018    0.000
```

Example

```
partSummary(out4, 5, c(1, 2, 15:24))

Latent Variables:
                   Estimate  Std.Err  z-value  P(>|z|)
  fXZ =~
    x1z1              0.367    0.053    6.902    0.000
    x1z2              0.434    0.056    7.715    0.000
    x1z3              0.441    0.053    8.300    0.000
    x2z1              0.550    0.056    9.788    0.000
    x2z2              0.616    0.062    9.970    0.000
    x2z3              0.519    0.057    9.115    0.000
    x3z1              0.504    0.059    8.604    0.000
    x3z2              0.628    0.063   10.039    0.000
    x3z3              0.535    0.059    9.128    0.000
```

Example

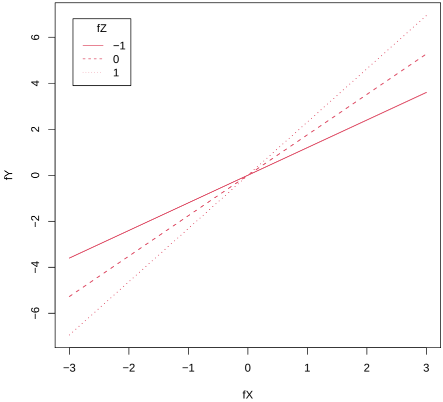
```
partSummary(out4, 6:7, -(10:35))

Regressions:
                   Estimate  Std.Err  z-value  P(>|z|)
  fY ~
    fX                1.757    0.270    6.515    0.000
    fZ               -0.111    0.105   -1.062    0.288
    fXZ               0.557    0.141    3.962    0.000

Covariances:
                   Estimate  Std.Err  z-value  P(>|z|)
  fX ~~
    fZ                0.232    0.058    3.987    0.000
    fXZ              -0.087    0.067   -1.297    0.195
  fZ ~~
    fXZ               0.040    0.066    0.613    0.540
```

Example

```
fitMeasures(out4, c("chisq", "df", "pvalue", "cfi", "tli", "rmsea", "srmr"))

  chisq      df  pvalue     cfi     tli   rmsea    srmr
134.186 111.000   0.066   0.993   0.991   0.020   0.030

probeOut4 <- probe2WayMC(fit      = out4,
                         nameX    = c("fX", "fZ", "fXZ"),
                         nameY    = "fY",
                         modVar   = "fZ",
                         valProbe = c(-1, 0, 1)
                         )

probeOut4$SimpleSlope

  fZ   est    se     z pvalue
1 -1 1.200 0.210 5.722      0
2  0 1.757 0.270 6.515      0
3  1 2.314 0.376 6.163      0
```

Example

```r
plotProbe(probeOut4, xlim = c(-3, 3), xlab = "fX", ylab = "fY")
```



Orthogonalization vs. Double Mean Centering

Orthogonalization and double mean centering tend to behave comparably, but each has its own strengths:

- When X and Z are bivariate normally distributed, both methods produce the same results.
- As X and/or Z stray from normality, orthogonalization produces biased estimates of the interaction effect, but double mean centering does not.
- Orthogonalization ensures that the latent XZ is perfectly independent of X and Z.
    - The X and Z parameters can be directly interpreted, without any conditioning

Example

We can also use the `indProd()` function from **semTools** to create the product indicators.

```
## Use semTools to orthogonalize:
dat2.2 <- indProd(data       = dat1,
                  var1       = c("x1", "x2", "x3"),
                  var2       = c("z1", "z2", "z3"),
                  match      = FALSE,
                  meanC      = FALSE,
                  doubleMC   = FALSE,
                  residualC  = TRUE,
                  namesProd  = colnames(products)
                  )

## Compare to our manual results:
all.equal(dat2[colnames(products)],
          dat2.2[colnames(products)],
          check.attributes = FALSE)

[1] TRUE
```

Example

```r
## Use semTools to double mean center:
dat3.2 <- indProd(data      = dat1,
                  var1      = c("x1", "x2", "x3"),
                  var2      = c("z1", "z2", "z3"),
                  match     = FALSE,
                  meanC     = TRUE,
                  doubleMC  = TRUE,
                  residualC = FALSE,
                  namesProd = colnames(products)
                  )

all.equal(dat3[colnames(products)],
          dat3.2[colnames(products)],
          check.attributes = FALSE)

[1] TRUE
```

Estimating Products of Latent Variables

We can directly estimate the interaction between two latent variables with the *latent moderated structural equations* (LMS) method.

- Introduced by Klein, Moosbrugger, Schermelleh-Engel, and Frank (1997) and formalized by Klein and Moosbrugger (2000)
- Currently only available in Mplus (via the `Xwith` command).
- Uses numerical integration to estimate the unobserved latent interaction term

Estimating Products of Latent Variables

LMS Strengths:

- Tends to perform the best out of all available methods
- No need to pre-process the data by manually computing product terms
- Pretty easy to implement if you have Mplus (see users guide for examples).

LMS Weaknesses:

- Only available in one (proprietary) software package
- Numerical integration is very slow and precludes calculation of most fit indices
- LMS does not work with categorical observed moderators

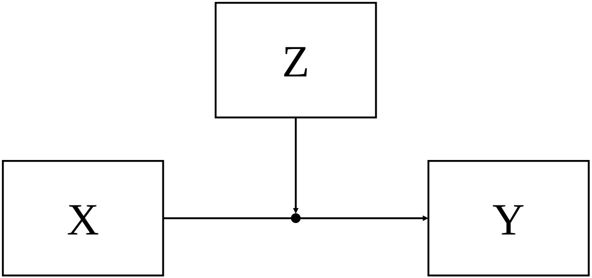
BSEM Example

```
mod <- "
fX =~ x1 + x2 + x3
fZ =~ z1 + z2 + z3
fY =~ y1 + y2 + y3

fY ~ fX + fZ + fX:fZ
"

out <- bsem(mod, data = dat1, std.lv = TRUE)

summary(out)
```

Multiple Moderation

Starting Point

So far, we've been looking at this type of model:



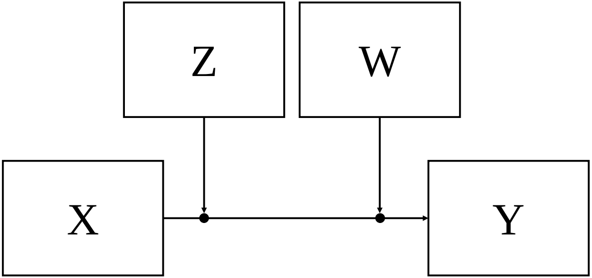
We've had one focal variable and one moderator.

- We've been asking questions about how the focal effect changes as a function of the moderator.
- There's no reason we need to restrict ourselves to a single moderator.

Multiple Moderation

Maybe we suspect that the focal effect changes as a function of two other variables.

- We could fit this type of model:



Now, the focal effect of X on Y changes as a function of both Z and W.

Multiple Moderation

The preceding diagram implies the following formula:

$$Y = \alpha + f(Z, W)X + \beta_2 Z + \beta_3 W + e,$$

Taking $f(Z, W)$ to be the following simple slope:

$$f(Z, W) = \beta_1 + \beta_4 Z + \beta_5 W$$

Produces the following analytic equation:

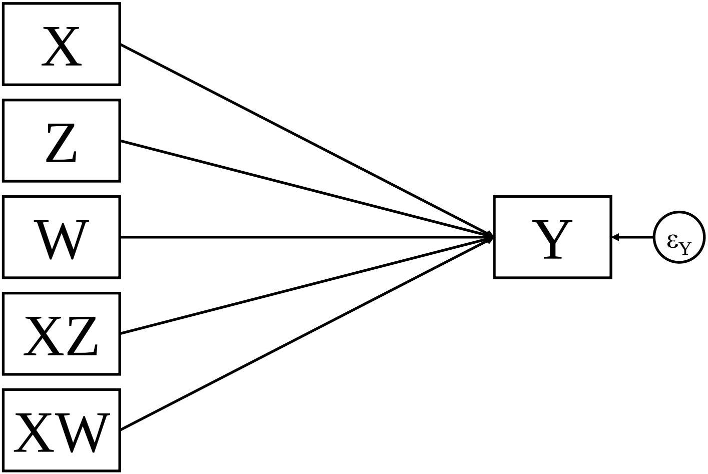
$$Y = \alpha + \beta_1 X + \beta_2 Z + \beta_3 W + \beta_4 XZ + \beta_5 XW + e$$

We can easily fit this model in any regression software

- We can test for significant moderating effects of Z and W by testing for non-zero β_4 and β_5, respectively.

Multiple Moderation

Our analytic diagram is predictably extended:



Example

Example

```
library(psych)
library(rockchalk)
dat1 <- readRDS("../data/bfiData1.rds")

## Additive model:
out1.1 <- lm(agree ~ conc + open + neuro, data = dat1)
summary(out1.1)


Call:
lm(formula = agree ~ conc + open + neuro, data = dat1)

Residuals:
     Min       1Q   Median       3Q      Max
-2.78733 -0.41707  0.09673  0.47476  2.12198

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  3.23373    0.12379  26.123  < 2e-16 ***
conc         0.06890    0.02647   2.603  0.00929 **
open         0.27661    0.02647  10.449  < 2e-16 ***
neuro       -0.10633    0.01205  -8.826  < 2e-16 ***
---
```

Moderated Moderation

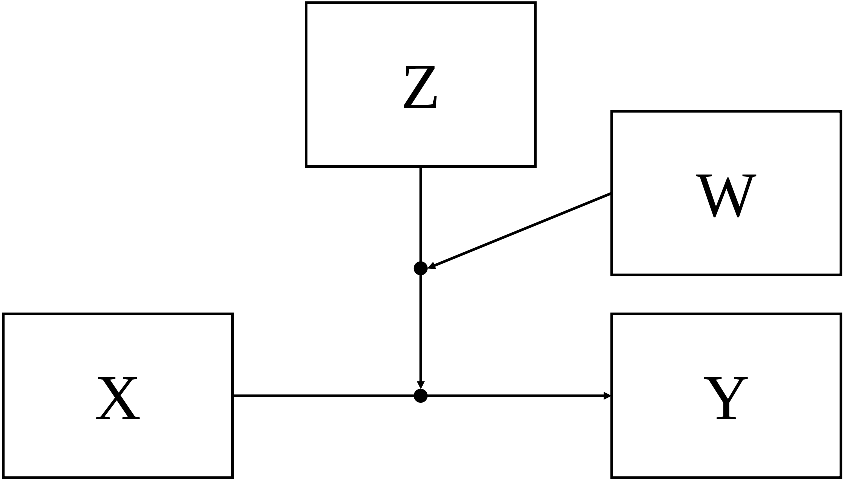
The additive two-way interaction model is more flexible than the simple single-moderator model, but it still imposes constraints.

- The moderating effect of Z (or W) on the $X \rightarrow Y$ relation is assumed to be constant across levels of W (or Z).
- I.e., the moderation is not moderated

We can relax this constraint by modeling moderation of the moderated effect using a three-way interaction.

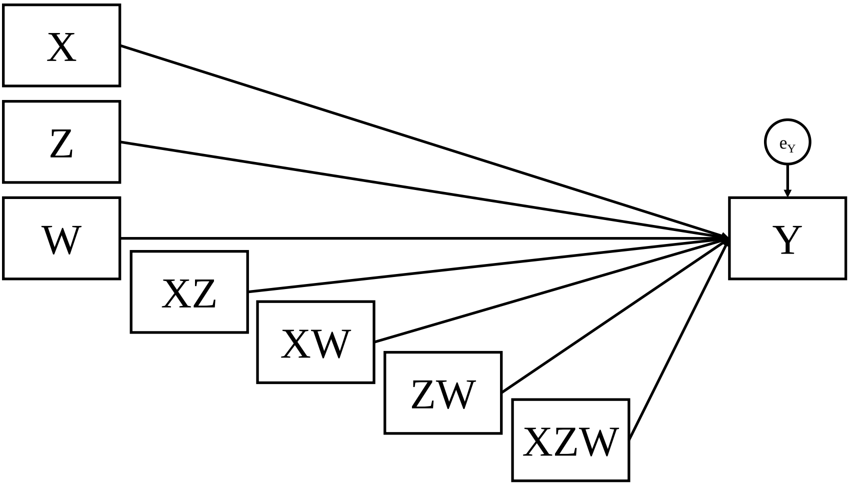
Moderated Moderation

Moderated moderation implies the following conceptual diagram:



Moderated Moderation

The preceding conceptual diagram implies this analytic diagram:



Moderated Moderation

The preceding diagram represents the following equation:

$$
\begin{aligned}
Y = \alpha + \beta_1 X + \beta_2 Z + \beta_3 W + \\
\beta_4 XZ + \beta_5 XW + \beta_6 ZW + \beta_7 XZW + e
\end{aligned}
$$

Which can be restructured into:

$$
\begin{aligned}
Y &= \alpha + (\beta_1 + \beta_4 Z + \beta_5 W + \beta_7 ZW)X + \\
&\quad \beta_2 Z + \beta_3 W + \beta_6 ZW + e \\
&= \alpha + g(Z, W)X + \beta_2 Z + \beta_3 W + \beta_6 ZW + e
\end{aligned}
$$

With moderated moderation, the simple slope is given by:

$$g(Z, W) = \beta_1 + \beta_4 Z + \beta_5 W + \beta_7 ZW$$

Which has the same structure as a single moderator model.

- Three-way simple slopes represent the moderated effect of Z on the $X \rightarrow Y$ relation at conditional values of W.

Example

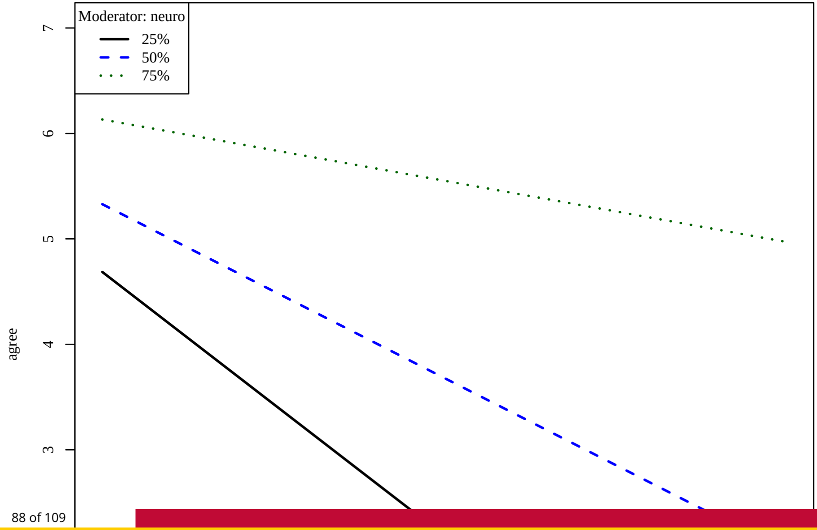
Example

```
## Three-way interaction model:
out1.3 <- lm(agree ~ open*conc*neuro, data = dat1)
summary(out1.3)


Call:
lm(formula = agree ~ open * conc * neuro, data = dat1)

Residuals:
     Min       1Q   Median       3Q      Max
-2.79789 -0.41779  0.09925  0.47556  2.10928

Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)     -0.58747    0.96633  -0.608  0.54328
open             1.27903    0.25747   4.968 7.23e-07 ***
conc             1.20831    0.26559   4.550 5.63e-06 ***
neuro            0.73766    0.32240   2.288  0.02222 *
open:conc       -0.29722    0.06935  -4.286 1.89e-05 ***
open:neuro      -0.21616    0.08091  -2.672  0.00760 **
conc:neuro      -0.25632    0.08244  -3.109  0.00190 **
open:conc:neuro  0.06541    0.02028   3.225  0.00128 **
---
```

Example

```r
par(family = "serif", cex = 0.75)
plotOut1.5 <- plotSlopes(model = out1.5,
                         plotx = "openXconc",
                         modx = "neuro",
                         plotPoints = FALSE,
                         modxVals =
                         quantile(dat1$neuro,
                                  c(0.25, 0.5, 0.75),
                                  na.rm = TRUE)
                         )
```

Example



CATEGORICAL MODERATORS

Categorical Moderators

Categorical moderators encode *group-specific* effects.

- E.g., if we include *sex* as a moderator, we are modeling separate focal effects for males and females.

Given a set of codes representing our moderator, we specify the interactions as before:

$$Y_{total} = \beta_0 + \beta_1 X_{inten} + \beta_2 Z_{male} + \beta_3 X_{inten} Z_{male} + \varepsilon$$

$$\begin{aligned} Y_{total} = \beta_0 &+ \beta_1 X_{inten} + \beta_2 Z_{lo} + \beta_3 Z_{mid} + \beta_4 Z_{hi} \\ &+ \beta_5 X_{inten} Z_{lo} + \beta_6 X_{inten} Z_{mid} + \beta_7 X_{inten} Z_{hi} + \varepsilon \end{aligned}$$

Example

```
## Load data:
socSup <- readRDS(paste0(dataDir, "social_support.rds"))

## Focal effect:
out3 <- lm(bdi ~ tanSat, data = socSup)
partSummary(out3, -c(1, 2))

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  24.4089     5.3502   4.562 1.54e-05
tanSat       -0.8100     0.3124  -2.593   0.0111

Residual standard error: 9.278 on 93 degrees of freedom
Multiple R-squared:  0.06742,Adjusted R-squared:  0.05739
F-statistic: 6.723 on 1 and 93 DF,  p-value: 0.01105
```

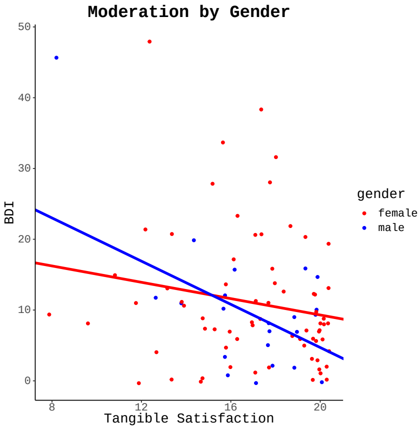
Example

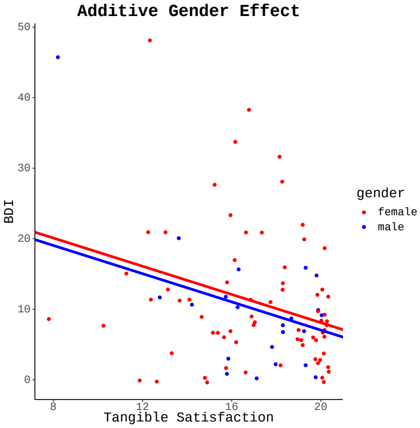
```
## Estimate the interaction:
out4 <- lm(bdi ~ tanSat * sex, data = socSup)
partSummary(out4, -c(1, 2))

Coefficients:
               Estimate Std. Error t value Pr(>|t|)
(Intercept)     20.8478     6.2114   3.356  0.00115
tanSat          -0.5772     0.3614  -1.597  0.11372
sexmale         14.3667    12.2054   1.177  0.24223
tanSat:sexmale  -0.9482     0.7177  -1.321  0.18978

Residual standard error: 9.267 on 91 degrees of freedom
Multiple R-squared:  0.08955,Adjusted R-squared:  0.05954
F-statistic: 2.984 on 3 and 91 DF,  p-value: 0.03537
```

Visualizing Categorical Moderation

$$\hat{Y}_{BDI} = 20.85 - 0.58X_{tsat} + 14.37Z_{male} - 0.95X_{tsat}Z_{male}$$

$$\hat{Y}_{BDI} = 28.10 - 1.00X_{tsat} - 1.05Z_{male}$$





Categorical Variable Moderation

When the moderator is a categorical variable, moderation implies between-group differences in the focal effect.

- This simplifies probing considerably
- The simple slopes are given (almost) directly in the output

Recall the simple slope formula:

$$SS = \beta_1 + \beta_3 Z$$

Because Z is a dummy code, this formula reduces to:

$$SS = \beta_1, \text{ or}$$
$$SS = \beta_1 + \beta_3$$

Example

Example

```
## Marginal focal effect:
out2.1 <- lm(conc ~ neuro, data = dat1)
summary(out2.1)


Call:
lm(formula = conc ~ neuro, data = dat1)

Residuals:
     Min       1Q   Median       3Q      Max
-2.55547 -0.33353  0.00824  0.36098  1.85381

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 3.437327   0.029659  115.90   <2e-16 ***
neuro       0.118144   0.008844   13.36   <2e-16 ***
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.533 on 2550 degrees of freedom
Multiple R-squared:  0.0654,Adjusted R-squared:  0.06504
F-statistic: 178.4 on 1 and 2550 DF,  p-value: < 2.2e-16
```

Example

```
summary(out2.2)


Call:
lm(formula = conc ~ neuro * educ, data = dat1)

Residuals:
     Min       1Q   Median       3Q      Max
-2.52324 -0.34119  0.01457  0.36247  1.86213

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  3.72924    0.10864  34.326  < 2e-16 ***
neuro        0.01259    0.03156   0.399 0.689990
educ2       -0.32892    0.11497  -2.861 0.004258 **
educ3       -0.30738    0.12102  -2.540 0.011146 *
neuro:educ2  0.11033    0.03346   3.297 0.000990 ***
neuro:educ3  0.12755    0.03552   3.591 0.000336 ***
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.5308 on 2546 degrees of freedom
Multiple R-squared:  0.0746,Adjusted R-squared:  0.07278
F-statistic: 41.05 on 5 and 2546 DF,  p-value: < 2.2e-16
```

Example

```
summary(out2.3)


Call:
lm(formula = conc ~ neuro * educ2, data = dat1)

Residuals:
     Min       1Q   Median       3Q      Max
-2.52324 -0.34119  0.01457  0.36247  1.86213

Coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)         3.40032    0.03761  90.401  < 2e-16 ***
neuro               0.12293    0.01111  11.063  < 2e-16 ***
educ2subHS          0.32892    0.11497   2.861  0.00426 **
educ2college        0.02154    0.06525   0.330  0.74134
neuro:educ2subHS   -0.11033    0.03346  -3.297  0.00099 ***
neuro:educ2college  0.01722    0.01972   0.873  0.38277
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.5308 on 2546 degrees of freedom
Multiple R-squared:  0.0746,Adjusted R-squared:  0.07278
F-statistic: 41.05 on 5 and 2546 DF,  p-value: < 2.2e-16
```

Example

```
summary(out2.4)


Call:
lm(formula = conc ~ neuro * educ3, data = dat1)

Residuals:
     Min       1Q   Median       3Q      Max 
-2.52324 -0.34119  0.01457  0.36247  1.86213 

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)
(Intercept)            3.42186    0.05331  64.183  < 2e-16
neuro                  0.14014    0.01629   8.601  < 2e-16
educ3subHS             0.30738    0.12102   2.540 0.011146
educ3highSchool       -0.02154    0.06525  -0.330 0.741340
neuro:educ3subHS      -0.12755    0.03552  -3.591 0.000336
neuro:educ3highSchool -0.01722    0.01972  -0.873 0.382770

(Intercept)           ***
neuro                 ***
educ3subHS            *  
educ3highSchool          
neuro:educ3subHS      ***
neuro:educ3highSchool    
---
Signif. codes:  
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.5308 on 2546 degrees of freedom
Multiple R-squared:  0.0746,Adjusted R-squared:  0.07278 
F-statistic: 41.05 on 5 and 2546 DF,  p-value: < 2.2e-16
```

Moderation via Multiple Group SEM

When our moderator is a categorical variable, we can use multiple group CFA/SEM to test for moderation.

- Categorical moderators define groups
- Significant moderation with categorical moderators implies between-group differences in the focal effect
- These hypotheses are easily tested with multiple group SEM

Whiteboard Time!

Example

```r
library(lavaan)
library(semTools)
dat2 <- readRDS("../data/bfiData2.rds")

## Multiple group moderation:
mod1 <- "
conc =~ C1 + C2 + C3 + C4 + C5
neuro =~ N1 + N2 + N3 + N4 + N5
"
```

Example

```
fit1 <- measurementInvariance(mod1,
                              data = dat2,
                              group = "educ",
                              std.lv = TRUE)

Warning:  The measurementInvariance function is deprecated, and it will cease to be
included in future versions of semTools.  See help('semTools-deprecated) for
details.
Error in measurementInvariance(mod1, data = dat2, group = "educ", std.lv = TRUE):
all lavaan() and lavOptions() arguments must named, including the "model="
argument.
```

Example

```
mod2 <- "
conc =~ C1 + C2 + C3 + C4 + C5
neuro =~ N1 + N2 + N3 + N4 + N5

conc ~ neuro

conc ~~ c(1.0, NA, NA)*conc
neuro ~~ c(1.0, NA, NA)*neuro

conc ~ c(0.0, NA, NA)*1.0
neuro ~ c(0.0, NA, NA)*1.0
"
```

Example

```
fit2 <- lavaan(mod2,
               data = dat2,
               std.lv = FALSE,
               auto.fix.first = FALSE,
               auto.var = TRUE,
               int.ov.free = TRUE,
               group = "educ",
               group.equal = c("loadings", "intercepts")
               )
```

Example

```
summary(fit2)

lavaan 0.6-11 ended normally after 78 iterations

  Estimator                                         ML
  Optimization method                           NLMINB
  Number of model parameters                       101
  Number of equality constraints                    40

  Number of observations per group:
    highSchool                                    1536
    subHS                                          192
    college                                        824

Model Test User Model:

  Test statistic                              1131.438
  Degrees of freedom                               134
  P-value (Chi-square)                           0.000
  Test statistic for each group:
    highSchool                                 573.290
    subHS                                      108.925
    college                                    449.224
```

Probing Multiple Group Moderation

Several advantages to testing moderation with multiple group SEM

- Remove measurement error from the estimates
- Test for factorial invariance
- *All information needed to plot/probe the simple slopes is contained directly in the output from the unrestricted model*

Example

Example

```
summary(fit2)

lavaan 0.6-11 ended normally after 78 iterations

  Estimator                                         ML
  Optimization method                           NLMINB
  Number of model parameters                       101
  Number of equality constraints                    40

  Number of observations per group:
    highSchool                                    1536
    subHS                                          192
    college                                        824

Model Test User Model:

  Test statistic                              1131.438
  Degrees of freedom                               134
  P-value (Chi-square)                           0.000
  Test statistic for each group:
    highSchool                                 573.290
    subHS                                      108.925
    college                                    449.224
```

References

Klein, A., & Moosbrugger, H. (2000). Maximum likelihood estimation of latent interaction effects with the LMS method. *Psychometrika*, *65*(4), 457–474.

Klein, A., Moosbrugger, H., Schermelleh-Engel, K., & Frank, D. (1997). A new approach to the estimation of latent interaction effects in structural equation models. *SoftStat*, *97*, 479–486.

Lin, G.-C., Wen, Z., Marsh, H. W., & Lin, H.-S. (2010). Structural equation models of latent interactions: Clarification of orthogonalizing and double-mean-centering strategies. *Structural Equation Modeling*, *17*(3), 374–391.

Little, T. D., Bovaird, J. A., & Widaman, K. F. (2006). On the merits of orthogonalizing powered and product terms: Implications for modeling interactions among latent variables. *Structural Equation Modeling*, *13*(4), 497–519.