Moderation

Introduction to SEM with Lavaan

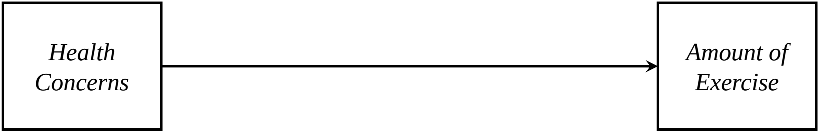
Kyle M. Lang

Department of Methodology & Statistics
Utrecht University

Outline

Refresher: Focal Effect Only

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

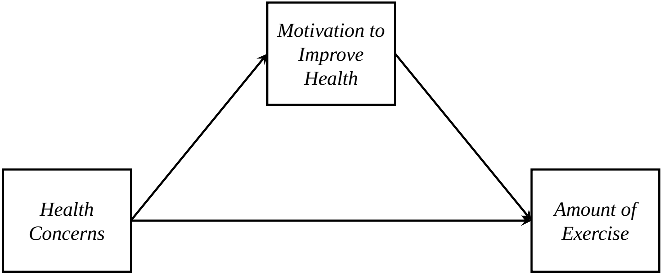


- Mediation, moderation, and conditional process analysis all attempt to describe the focal effect in more detail.
- We always begin by hypothesizing a focal effect.

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*).

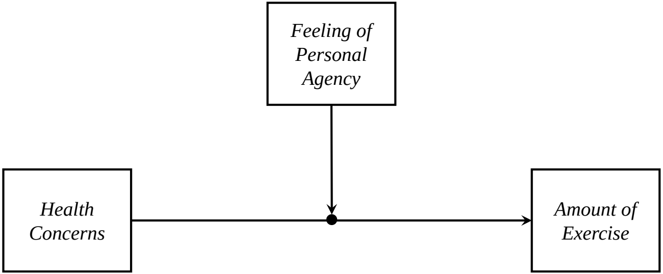


Refresher: Moderation Hypothesis

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:

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

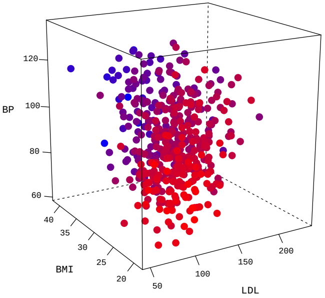
This additive equation assumes that X and Z are independent predictors of Y.

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

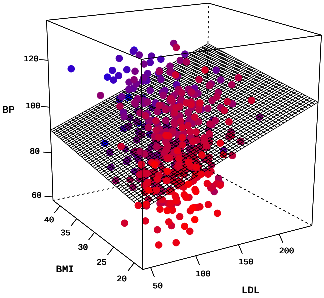
- X and Z *can* be correlated.
- β_1 and β_2 are *partial* regression coefficients.
- The effect of X on Y is the same at **all levels** of Z, and the effect of Z on Y is the same at **all levels** of X.

Additive Regression

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

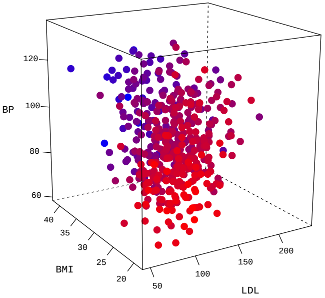


$\longrightarrow$

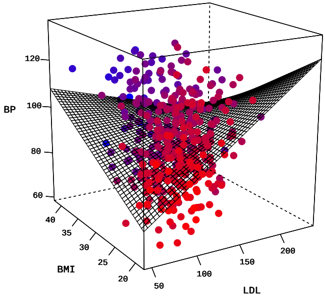


Moderated Regression

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



$\longrightarrow$



Equations

The following derivation is adapted from Hayes (2022).

- 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 Hayes (2022).

- 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

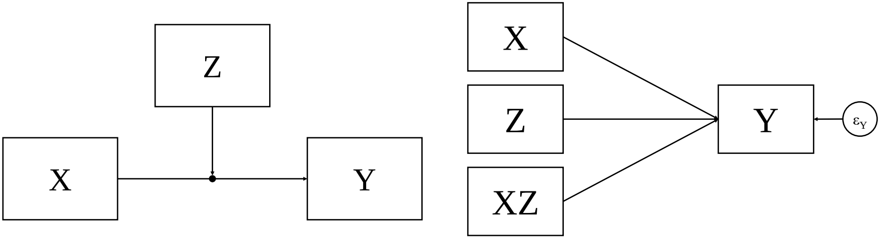
- Substituting Equation 2 into Equation 1 leads to:

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

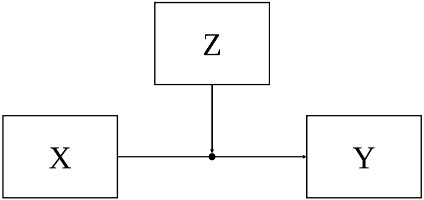
- Which, after distributing X and reordering terms, becomes:

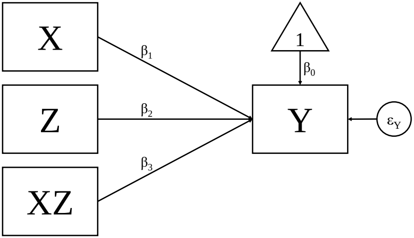
$$Y = \beta_0 + \beta_1 X + \beta_2 Z + \beta_3 XZ + \varepsilon$$

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:

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

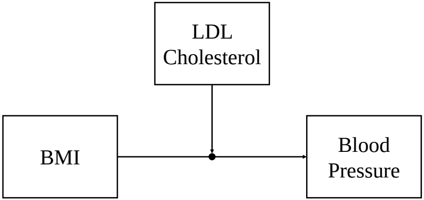
- $\hat{\beta}_3$ quantifies the effect of Z on the focal effect (the $X \rightarrow Y$ effect).
  - For a unit change in Z, $\hat{\beta}_3$ is the expected change in the effect of X on Y.
- $\hat{\beta}_1$ and $\hat{\beta}_2$ are *conditional effects*.
  - Interpreted where the other predictor is zero.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

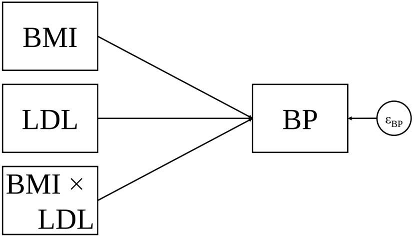
Looking at the *diabetes* dataset.

- 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

```
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

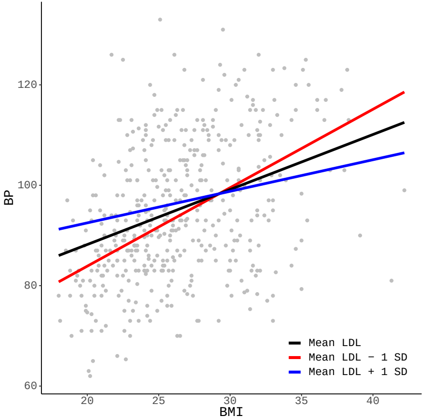
```
## Additive Model:
out1 <- lm(bp ~ bmi + ldl, data = dDat)
partSummary(out1, -c(1, 2))

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'

## Estimate the model:
lavOut <- sem(mod, data = dDat)
```

Example

```
partSummary(lavOut, 7:9)

Regressions:
                   Estimate   Std.Err  z-value  P(>|z|)
  bp ~
    bmi               2.868     0.539    5.322    0.000
    ldl               0.449     0.127    3.545    0.000
    bmi:ldl          -0.015     0.005   -3.270    0.001

Intercepts:
                   Estimate   Std.Err  z-value  P(>|z|)
   .bp               14.481    14.227    1.018    0.309

Variances:
                   Estimate   Std.Err  z-value  P(>|z|)
   .bp              155.871    10.485   14.866    0.000
```

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 not much 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:

$$SE_{f(Z)} = \sqrt{SE_{\beta_1}^2 + 2Z \cdot cov(\beta_1, \beta_3) + Z^2 SE_{\beta_3}^2}$$

So, you can test the significance of a simple slope by constructing a t-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

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

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

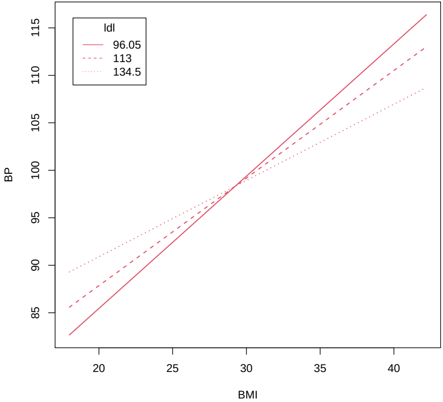
```
library(semTools)

## 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))
                     )
```

Example

```
## View the results:
ssOut

$SimpleIntcept
       ldl    est    se      z pvalue
25%  96.05 57.585 4.017 14.334      0
50% 113.00 65.192 3.736 17.449      0
75% 134.50 74.840 4.944 15.139      0

$SimpleSlope
       ldl   est    se     z pvalue
25%  96.05 1.393 0.156 8.942      0
50% 113.00 1.133 0.140 8.107      0
75% 134.50 0.803 0.178 4.508      0
```

Example

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



Latent Variable Interactions

Latent Variable Interactions

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

1. Construct a product term from the 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 one of the predictors is latent, things get more difficult.

- For observed and discrete moderators, use multiple group modeling.
- For continuous and/or latent moderators, 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 products as indicators of an 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.
- We'd also like to interpret our 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 Procedure

Say we think that Z moderates the $X \rightarrow Y$ effect.

- 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 Procedure

Say we think that Z moderates the $X \to Y$ effect.

- 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.

We perform the orthogonalization procedure as follows:

1. Construct all possible product terms: $\{x_1z_1, x_1z_2, x_1z_3, x_2z_1, x_2z_2, x_2z_3, x_3z_1, x_3z_2, x_3z_3\}$.

Orthogonalization Procedure

Say we think that Z moderates the $X \rightarrow Y$ effect.

- 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.

We perform the orthogonalization procedure as follows:

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 Procedure

3. Calculate each product term's residual:

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

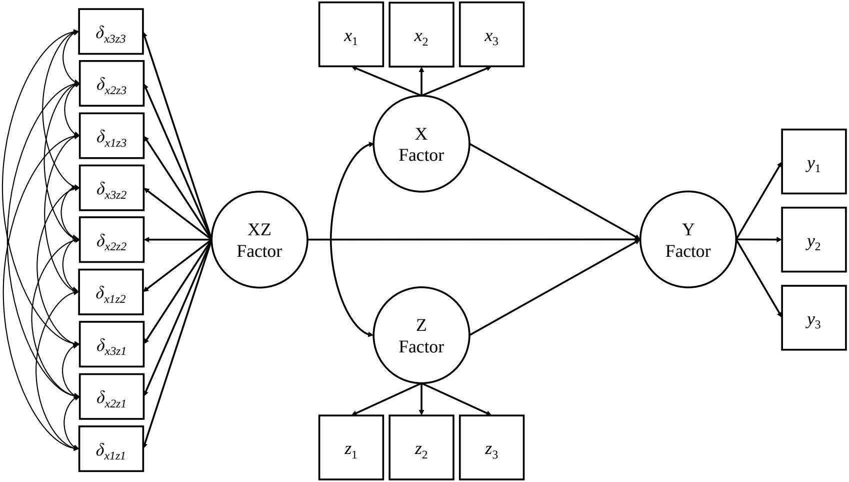
Orthogonalization Procedure

3. Calculate each product term's residual:

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

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

Orthogonalization Diagram



Example

First, we check the measurement model.

```
## Read in some data:
dat1 <- readRDS("../data/lecture12Data.rds")

## Define the CFA model:
mod1 <- '
fX =~ x1 + x2 + x3
fZ =~ z1 + z2 + z3
fY =~ y1 + y2 + y3
'

## Estimate the model:
out1 <- cfa(mod1, data = dat1, std.lv = TRUE)
```

Example

```
partSummary(out1, 1:6)

lavaan 0.6-12.1708 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, 7)

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, 8)

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, 9)

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.

```
## Define the additive structural model:
mod2 <- '
fX =~ x1 + x2 + x3
fZ =~ z1 + z2 + z3
fY =~ y1 + y2 + y3

fY ~ fX + fZ
'

## Fit the model:
out2 <- sem(mod2, data = dat1, std.lv = TRUE)
```

Example

```
partSummary(out2, 1:6)

lavaan 0.6-12.1708 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, 7)

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, 8:9)

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, 10)

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(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) # For data processing routines

## 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

```
## Estimate the model:
out3 <- sem(mod3, data = dat2, std.lv = TRUE, meanstructure = TRUE)

partSummary(out3, 7, 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, 7, 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

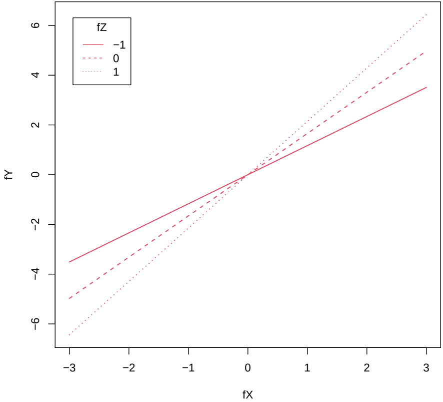
```
partSummary(out3, 8:9, -(14:39))

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

## Test simple slopes:
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 Procedure

We can also define the interaction factor with double mean centering.

1. Mean center every indicator of X and Z:

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

Double Mean Centering Procedure

We can also define the interaction factor with 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 Procedure

3. Mean center each product term:

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

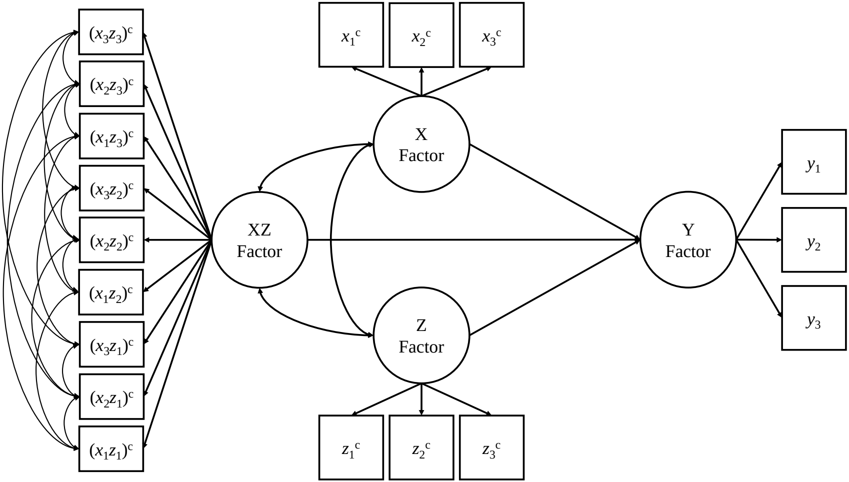
Double Mean Centering Procedure

3. Mean center each product term:

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

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 Diagram



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, 7, 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, 7, 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

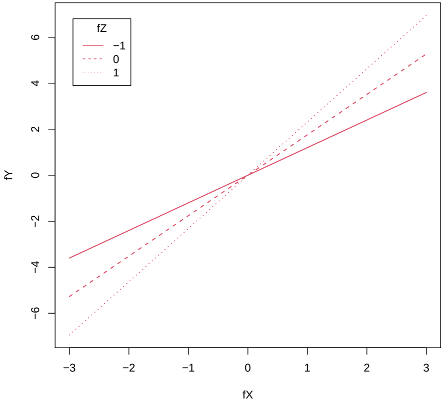
```
partSummary(out4, 8:9, -(10:35))

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

```
## Check model fit:
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

## Test simple slopes:
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

```
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

In theory, we can directly estimate a latent variable defined as the product of two (or more) other latent variables.

- We cannot use ordinary ML estimation methods.

Three approaches seem sensible.

1. Latent moderated structural equations (LMS; Klein & Moosbrugger, 2000; Klein, Moosbrugger, Schermelleh-Engel, & Frank, 1997).
2. Structural-after-measurement (SAM) estimation (Rosseel & Loh, in press).
3. Bayesian SEM.

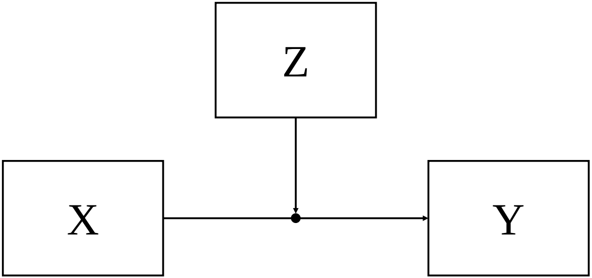
None of the above is currently implemented in **lavaan**.

- The SAM approach should be available soon.

Multiple Moderation

Starting Point

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



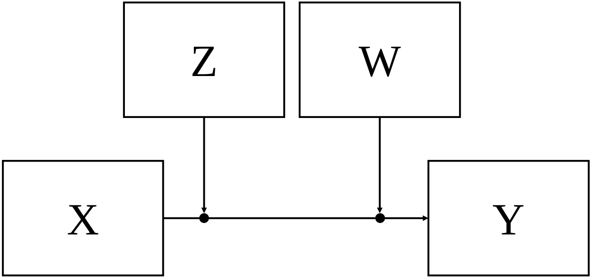
We've had one focal variable and one moderator.

- 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:

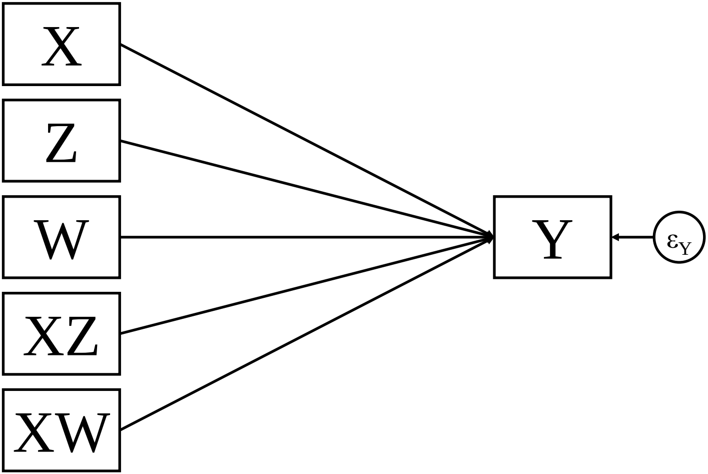
$$Y = \alpha + \beta_1 X + \beta_2 Z + \beta_3 W + \beta_4 XZ + \beta_5 XW + e$$

We can easily fit this model in any regression (or path modeling) 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:



Moderated Moderation

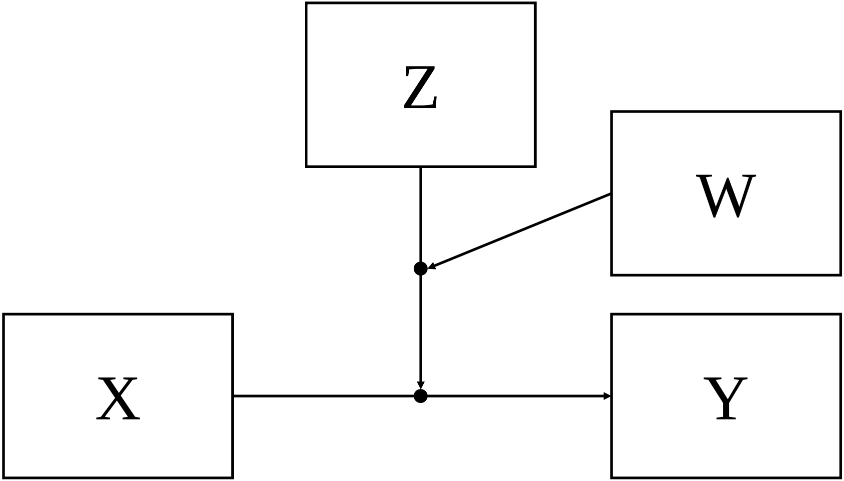
The additive two-way interaction model is more flexible than the simple single-moderator model, but it still imposes constraints.

- 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.

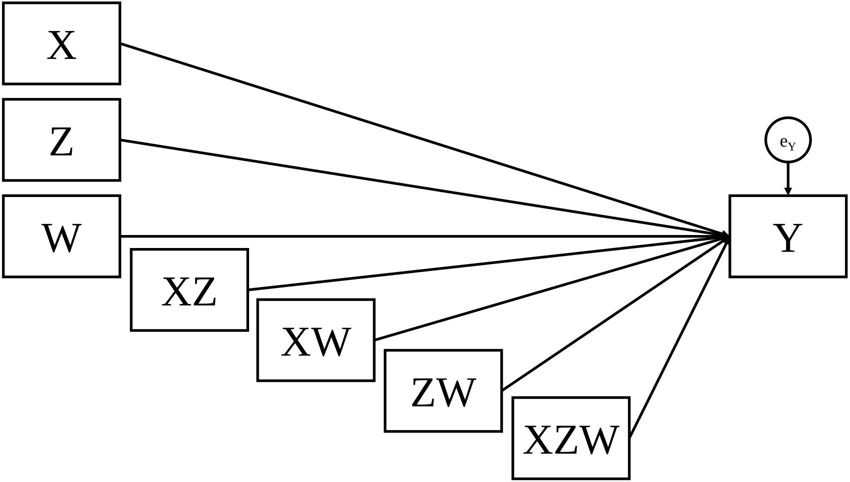
Moderated Moderation

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:

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

Which can be restructured into:

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

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.

Example

```r
## Read in the BFI data:
dat1 <- readRDS("../data/bfiData1.rds")

## Three-way interaction model:
mod <- '
agree ~ 1 + open + conc + neuro + open:conc + open:neuro + conc:neuro + ocn
'

## Create the 3-way interaction via a pipeline and estimate the model:
out <- dat1 %>%
    mutate(ocn = open * conc * neuro) %>%
    sem(mod, data = .)
```

Example

```
partSummary(out, 7:9)

Regressions:
                   Estimate  Std.Err  z-value  P(>|z|)
  agree ~
    open              1.279    0.257    4.975    0.000
    conc              1.208    0.265    4.557    0.000
    neuro             0.738    0.322    2.292    0.022
    open:conc        -0.297    0.069   -4.292    0.000
    open:neuro       -0.216    0.081   -2.676    0.007
    conc:neuro       -0.256    0.082   -3.114    0.002
    ocn               0.065    0.020    3.230    0.001

Intercepts:
                   Estimate  Std.Err  z-value  P(>|z|)
   .agree            -0.587    0.965   -0.609    0.543

Variances:
                   Estimate  Std.Err  z-value  P(>|z|)
   .agree             0.481    0.013   35.721    0.000
```

Example

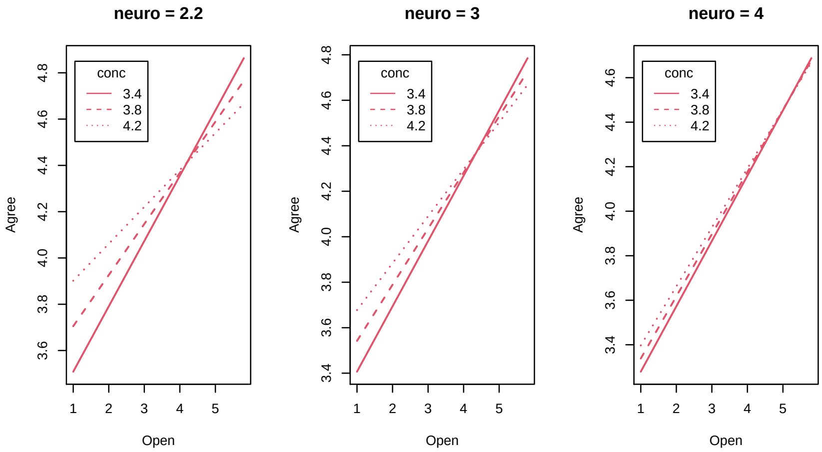
```r
## Compute simple slopes by conditioning on both moderators:
ssOut <- probe3WayMC(out,
                     nameX = c("open", "conc", "neuro",
                               "open:conc", "open:neuro", "conc:neuro",
                               "ocn"),
                     nameY = "agree",
                     modVar = c("conc", "neuro"),
                     valProbe1 = quantile(dat1$conc, c(0.25, 0.5, 0.75)),
                     valProbe2 = quantile(dat1$neuro, c(0.25, 0.5, 0.75))
                     )
```

Example

```
## View simple slopes:
ssOut$SimpleSlope

  conc neuro   est    se     z pvalue
1  3.4   2.2 0.282 0.034 8.316      0
2  3.8   2.2 0.221 0.035 6.400      0
3  4.2   2.2 0.160 0.042 3.802      0
4  3.4   3.0 0.287 0.031 9.346      0
5  3.8   3.0 0.247 0.027 8.980      0
6  4.2   3.0 0.206 0.032 6.379      0
7  3.4   4.0 0.293 0.041 7.179      0
8  3.8   4.0 0.279 0.032 8.607      0
9  4.2   4.0 0.265 0.033 8.032      0
```

Example

```
plotProbe(ssOut, xlim = range(dat1$open), xlab = "Open", ylab = "Agree")
```



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("../data/social_support.rds")

## Focal effect model:
mod <- 'bdi ~ 1 + tanSat'

## Fit the model and summarize the results:
sem(mod, data = socSup) %>% partSummary(7:9)

Regressions:
                   Estimate  Std.Err  z-value  P(>|z|)
  bdi ~
    tanSat           -0.810    0.309   -2.621    0.009

Intercepts:
                   Estimate  Std.Err  z-value  P(>|z|)
   .bdi              24.409    5.294    4.611    0.000

Variances:
                   Estimate  Std.Err  z-value  P(>|z|)
   .bdi              84.261   12.226    6.892    0.000
```

Example

```r
## Moderated model:
mod <- 'bdi ~ 1 + tanSat + male + tanSat:male'

## Dummy code sex in a pipeline and estimate the model:
out <- socSup %>%
    mutate(male = as.numeric(sex == "male")) %>%
    sem(mod, data = .)
```

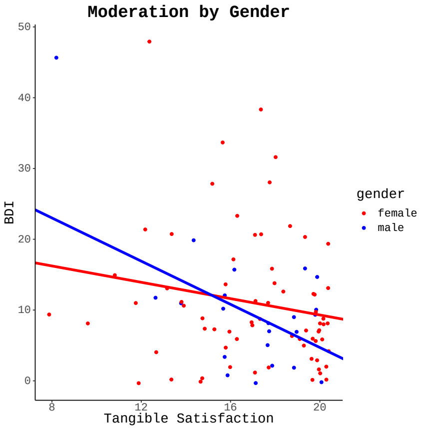
Example

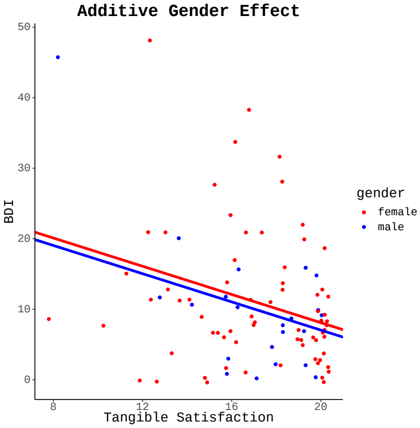
```
partSummary(out, 7:9)

Regressions:
                   Estimate  Std.Err  z-value  P(>|z|)
  bdi ~
    tanSat           -0.577    0.354   -1.632    0.103
    male             14.367   11.946    1.203    0.229
    tanSat:male      -0.948    0.702   -1.350    0.177

Intercepts:
                   Estimate  Std.Err  z-value  P(>|z|)
   .bdi              20.848    6.079    3.429    0.001

Variances:
                   Estimate  Std.Err  z-value  P(>|z|)
   .bdi              82.261   11.936    6.892    0.000
```

Visualizing Categorical Moderation

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

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





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.
- We can directly test these hypotheses with multiple group SEM.

Example

```r
## Read the data and subset to only high school and college graduates:
dat2 <- readRDS("../data/bfiData2.rds") %>%
    filter(educ %in% c("highSchool", "college"))

## Specify the (configurally invariance) measurement model:
mod0 <- '
agree =~ A1 + A2 + A3 + A4 + A5
open  =~ O1 + O2 + O3 + O4 + O5
'

## Estimate the unrestricted model:
out0 <- cfa(mod0, data = dat2, std.lv = TRUE, group = "educ")
```

Example

```r
## Define the weakly invariant model:
mod1 <- measEq.syntax(configural.model = out0,
                      group = "educ",
                      group.equal = "loadings") %>%
    as.character()

## Define the strongly invariant model:
mod2 <- measEq.syntax(configural.model = out0,
                      group = "educ",
                      group.equal = c("loadings", "intercepts")
                      ) %>%
    as.character()

## Estimate the models:
out1 <- cfa(mod1, data = dat2, group = "educ")
out2 <- cfa(mod2, data = dat2, group = "educ")

## Test measurement invariance:
compareFit(out0, out1, out2) %>% summary()
```

Example

```
################### Nested Model Comparison #########################
Chi-Squared Difference Test

     Df   AIC   BIC  Chisq Chisq diff Df diff Pr(>Chisq)
out0 68 75336 75694 418.25
out1 76 75358 75669 455.70     37.451       8  9.505e-06 ***
out2 84 75461 75726 575.15    119.446       8  < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

####################### Model Fit Indices ###########################
        chisq df pvalue rmsea   cfi   tli  srmr         aic         bic
out0 418.253† 68   .000 .066  .906† .875  .046† 75336.292† 75693.810
out1 455.705  76   .000 .065† .898  .879† .050  75357.744  75669.130†
out2 575.151  84   .000 .070  .868  .858  .056  75461.190  75726.445

################## Differences in Fit Indices #######################
            df  rmsea    cfi    tli  srmr     aic     bic
out1 - out0  8 -0.001 -0.008  0.004 0.004  21.451 -24.680
out2 - out1  8  0.005 -0.030 -0.021 0.006 103.446  57.315
```

Example

```r
## Specify a structural model:
mod3 <- '
agree =~ A1 + A2 + A3 + A4 + A5
open  =~ O1 + O2 + O3 + O4 + O5

agree ~ open
'

## Estimate the model with strong invariance constraints:
out3 <- sem(mod3,
            dat = dat2,
            std.lv = TRUE,
            group = "educ",
            group.equal = c("loadings", "intercepts")
            )
```

Example

```
## Check the group-specific slopes:
partSummary(out3, c(8, 10, 14, 16))

Group 1 [highSchool]:

Regressions:
                   Estimate  Std.Err  z-value  P(>|z|)
  agree ~
    open             -0.321    0.040   -7.957    0.000

Group 2 [college]:

Regressions:
                   Estimate  Std.Err  z-value  P(>|z|)
  agree ~
    open             -0.203    0.051   -3.972    0.000
```

Example

```
## Specify the restricted model:
mod4 <- '
agree =~ A1 + A2 + A3 + A4 + A5
open  =~ O1 + O2 + O3 + O4 + O5

agree ~ c(beta, beta) * open
'

## Estimate the model:
out4 <- sem(mod4,
            dat = dat2,
            std.lv = TRUE,
            group = "educ",
            group.equal = c("loadings", "intercepts")
            )
```

Example

```
## Check the slopes:
partSummary(out4, c(8, 10, 14, 16))

Group 1 [highSchool]:

Regressions:
                   Estimate  Std.Err  z-value  P(>|z|)
  agree ~
    open    (beta)   -0.278    0.032   -8.621    0.000

Group 2 [college]:

Regressions:
                   Estimate  Std.Err  z-value  P(>|z|)
  agree ~
    open    (beta)   -0.278    0.032   -8.621    0.000
```

Example

```
## Do a chi-squared difference test for moderation:
anova(out3, out4)

Chi-Squared Difference Test

     Df   AIC   BIC  Chisq Chisq diff Df diff Pr(>Chisq)
out3 84 75461 75726 575.15
out4 85 75463 75722 578.59      3.435       1    0.06383 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Example

```
## Do a similar test via OLS regression:
dat1 %>%
    filter(educ %in% c("highSchool", "college")) %$%
    lm(agree ~ open * educ) %>%
    partSummary(-(1:2))

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)       3.24965    0.12321  26.376   <2e-16
open              0.27115    0.03134   8.652   <2e-16
educcollege       0.03975    0.22849   0.174    0.862
open:educcollege -0.05654    0.05856  -0.965    0.334

Residual standard error: 0.6972 on 2356 degrees of freedom
Multiple R-squared:  0.05314,Adjusted R-squared:  0.05194
F-statistic: 44.08 on 3 and 2356 DF,  p-value: < 2.2e-16
```

Probing Multiple Group Moderation

Testing moderation with multiple group SEM has several advantages.

- Remove measurement error from the estimates
- Test for factorial invariance
- All simple effects are directly estimated in the unrestricted model

Simple Slopes & Intercepts

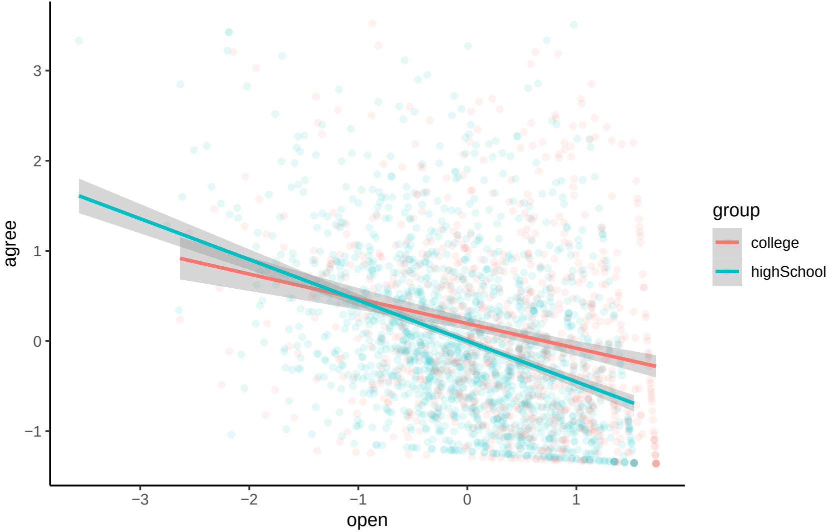
```
Group 1 [highSchool]:

Regressions:
                   Estimate  Std.Err  z-value  P(>|z|)
  agree ~
    open             -0.321    0.040   -7.957    0.000

Intercepts:
                   Estimate  Std.Err  z-value  P(>|z|)
   .agree             0.000

Group 2 [college]:

Regressions:
                   Estimate  Std.Err  z-value  P(>|z|)
  agree ~
    open             -0.203    0.051   -3.972    0.000

Intercepts:
                   Estimate  Std.Err  z-value  P(>|z|)
   .agree             0.170    0.056    3.058    0.002
```

Simple Slopes Visualized

We can visualize the simple slopes by plotting the factor scores.

```
library(ggplot2)

## Generate factor scores:
tmp <- predict(out3)

## Stack factor scores into a "tidy" dataset:
pData <- data.frame(do.call(rbind, tmp),
                    group = rep(names(tmp), sapply(tmp, nrow))
                    )

## Create a simple slopes plot:
ssPlot <- ggplot(pData, aes(open, agree, color = group)) +
    geom_point(alpha = 0.1) +
    geom_smooth(method = "lm") +
    theme_classic()
```

Simple Slopes Visualized



References

Hayes, A. F. (2022). *Introduction to mediation, moderation, and conditional process analysis: A regression-based approach* (3rd ed.). New York: Guilford Press.

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.

References

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.

Rosseel, Y., & Loh, W. W. (in press). A structural-after-measurement (SAM) approach to structural equation modeling. *Psychological Methods*. Retrieved from `https://osf.io/pekbm/`