Lecture 12: Loose Ends

Kyle M. Lang

Institute for Measurement, Methodology, Analysis & Policy
Texas Tech University
Lubbock, TX

2016 Stats Camp

Outline

- Latent variable interactions
- Moderated logistic regression
- Effect size for conditional process analysis

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

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

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

Computing Interaction Indicators

The alternative to the LMS-type 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:

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

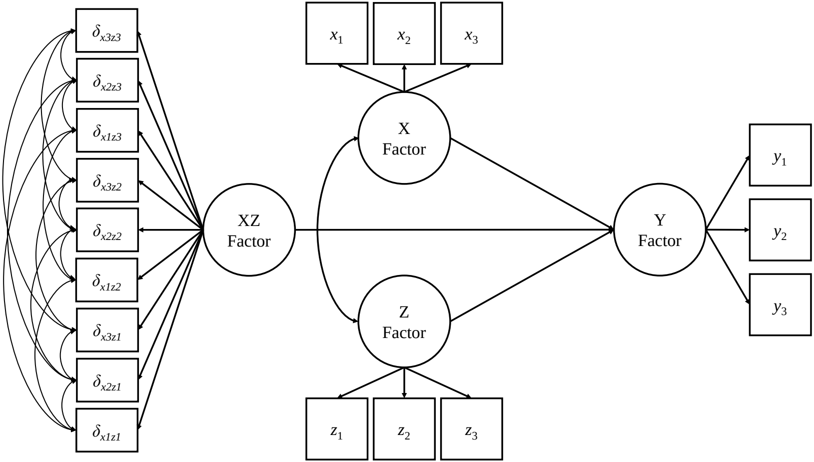
Orthogonalization

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 as represented in the following figure.

Orthogonalization



Example

```
library(lavaan)
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)
summary(out1)
```

```
lavaan (0.5-20) converged normally after  17 iterations

  Number of observations                           500

  Estimator                                         ML
  Minimum Function Test Statistic               41.021
  Degrees of freedom                                24
  P-value (Chi-square)                           0.017

Parameter Estimates:

  Information                                 Expected
```

Example

```
  Standard Errors                                  Standard

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

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 ~~
```

Example

```
    fY                0.156    0.057    2.739    0.006

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

```
round(fitMeasures(out1)[c("chisq", "df", "pvalue", "cfi",
                          "tli", "rmsea", "srmr")], 3)
```

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

Example

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

fY ~ fX + fZ
"
out2 <- sem(mod2, data = dat1, std.lv = TRUE)
summary(out2)
```

```
lavaan (0.5-20) converged normally after  22 iterations

  Number of observations                           500

  Estimator                                         ML
  Minimum Function Test Statistic               41.021
  Degrees of freedom                                24
  P-value (Chi-square)                           0.017

Parameter Estimates:

  Information                                 Expected
```

Example

```
  Standard Errors                             Standard

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

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
```

Example

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

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

```
round(fitMeasures(out2)[c("chisq", "df", "pvalue", "cfi",
                         "tli", "rmsea", "srmr")], 3)
```

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

Example

```r
predDat <- as.matrix(dat1[ , -grep("y", colnames(dat1))])
dat2 <- dat1
## Construct product terms:
x1z1 <- with(dat2, x1*z1)
x1z2 <- with(dat2, x1*z2)
x1z3 <- with(dat2, x1*z3)
x2z1 <- with(dat2, x2*z1)
x2z2 <- with(dat2, x2*z2)
x2z3 <- with(dat2, x2*z3)
x3z1 <- with(dat2, x3*z1)
x3z2 <- with(dat2, x3*z2)
x3z3 <- with(dat2, x3*z3)
## Residualize the product terms:
dat2$x1z1R <- lm(x1z1 ~ predDat)$resid
dat2$x1z2R <- lm(x1z2 ~ predDat)$resid
dat2$x1z3R <- lm(x1z3 ~ predDat)$resid
dat2$x2z1R <- lm(x2z1 ~ predDat)$resid
dat2$x2z2R <- lm(x2z2 ~ predDat)$resid
dat2$x2z3R <- lm(x2z3 ~ predDat)$resid
dat2$x3z1R <- lm(x3z1 ~ predDat)$resid
dat2$x3z2R <- lm(x3z2 ~ predDat)$resid
dat2$x3z3R <- lm(x3z3 ~ predDat)$resid
```

Example

```
mod3 <- "
fX =~ x1 + x2 + x3
fZ =~ z1 + z2 + z3
fY =~ y1 + y2 + y3
fXZ =~ x1z1R + x1z2R + x1z3R +
x2z1R + x2z2R + x2z3R +
x3z1R + x3z2R + x3z3R

fY ~ fX + fZ + fXZ

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

x1z1R ~~ x1z2R + x1z3R + x2z1R + x3z1R
x1z2R ~~ x1z3R + x2z2R + x3z2R
x1z3R ~~ x2z3R + x3z3R

x2z1R ~~ x2z2R + x2z3R + x3z1R
x2z2R ~~ x2z3R + x3z2R
x2z3R ~~ x3z3R
```

Example

```
x3z1R ~~ x3z2R + x3z3R
x3z2R ~~ x3z3R
"
out3 <- sem(mod3, data = dat2, std.lv = TRUE)
summary(out3)
```

```
lavaan (0.5-20) converged normally after  53 iterations

  Number of observations                           500

  Estimator                                         ML
  Minimum Function Test Statistic               74.899
  Degrees of freedom                               113
  P-value (Chi-square)                           0.998

Parameter Estimates:

  Information                                 Expected
  Standard Errors                             Standard

Latent Variables:
                   Estimate  Std.Err  Z-value  P(>|z|)
```

Example

```
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
fXZ =~
  x1z1R         0.361    0.053     6.833    0.000
  x1z2R         0.427    0.056     7.615    0.000
  x1z3R         0.432    0.053     8.190    0.000
  x2z1R         0.558    0.056     9.914    0.000
  x2z2R         0.616    0.062    10.008    0.000
  x2z3R         0.520    0.057     9.153    0.000
  x3z1R         0.516    0.059     8.805    0.000
  x3z2R         0.626    0.063    10.007    0.000
  x3z3R         0.521    0.058     8.936    0.000
```

Example

```
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 ~~
    fZ                0.232    0.058    3.987    0.000
    fXZ               0.000
  fZ ~~
    fXZ               0.000
  x1z1R ~~
    x1z2R             0.273    0.032    8.397    0.000
    x1z3R             0.309    0.033    9.358    0.000
    x2z1R             0.232    0.031    7.566    0.000
    x3z1R             0.235    0.032    7.376    0.000
  x1z2R ~~
    x1z3R             0.231    0.032    7.243    0.000
    x2z2R             0.211    0.035    5.982    0.000
    x3z2R             0.250    0.041    6.163    0.000
```

Example

```
  x1z3R ~~
    x2z3R              0.213    0.030    7.010    0.000
    x3z3R              0.213    0.034    6.312    0.000
  x2z1R ~~
    x2z2R              0.247    0.043    5.787    0.000
    x2z3R              0.252    0.040    6.368    0.000
    x3z1R              0.233    0.033    7.103    0.000
  x2z2R ~~
    x2z3R              0.304    0.043    7.086    0.000
    x3z2R              0.199    0.040    5.018    0.000
  x2z3R ~~
    x3z3R              0.139    0.030    4.570    0.000
  x3z1R ~~
    x3z2R              0.212    0.041    5.116    0.000
    x3z3R              0.260    0.040    6.454    0.000
  x3z2R ~~
    x3z3R              0.157    0.041    3.846    0.000

Variances:
                    Estimate  Std.Err  Z-value  P(>|z|)
    x1                 0.511    0.042   12.093    0.000
    x2                 0.514    0.042   12.221    0.000
    x3                 0.548    0.046   11.977    0.000
```

Example

z1	0.523	0.052	10.142	0.000
z2	0.546	0.052	10.444	0.000
z3	0.461	0.048	9.704	0.000
y1	0.495	0.043	11.398	0.000
y2	0.542	0.044	12.334	0.000
y3	0.444	0.040	11.179	0.000
x1z1R	0.743	0.050	14.912	0.000
x1z2R	0.754	0.055	13.682	0.000
x1z3R	0.694	0.050	13.824	0.000
x2z1R	0.641	0.057	11.332	0.000
x2z2R	0.708	0.067	10.575	0.000
x2z3R	0.671	0.056	12.009	0.000
x3z1R	0.736	0.060	12.310	0.000
x3z2R	0.724	0.070	10.277	0.000
x3z3R	0.707	0.060	11.823	0.000
fX	1.000			
fZ	1.000			
fY	1.000			
fXZ	1.000			

Example

```
round(fitMeasures(out3)[c("chisq", "df", "pvalue", "cfi",
                          "tli", "rmsea", "srmr")], 3)
```

```
 chisq      df  pvalue     cfi     tli   rmsea    srmr
74.899 113.000   0.998   1.000   1.015   0.000   0.020
```

```
library(semTools)
out3.2 ←
    sem(mod3, data = dat2, std.lv = TRUE, meanstructure =
        TRUE)
probeOut3 ← probe2WayRC(fit = out3.2,
                        nameX = c("fX", "fZ", "fXZ"),
                        nameY = "fY",
                        modVar = "fZ",
                        valProbe = c(-1, 0, 1)
                        )
probeOut3$SimpleSlope
```

Example

```
      fZ    Slope        SE       Wald p
[1,] -1 1.169652 0.1572049  7.440306 0
[2,]  0 1.657530 0.0979130 16.928605 0
[3,]  1 2.145409 0.1426381 15.040921 0
```

Matched Pair Variation

If you are willing to assume exchangeable indicators (i.e., *essential tau equivalence*), then you don't need to compute all possible interaction terms.

The so-called *matched pair* strategy suggests constructing only three product variables (when each first order construct has three indicators).

- Each product variable is simply constructed from paired indicators of the two first-order constructs:

$$x_1 z_1 = x_1 \times z_1$$
$$x_2 z_2 = x_2 \times z_2$$
$$x_3 z_3 = x_3 \times z_3$$

Example

```
mod4 <- "
fX =~ x1 + x2 + x3
fZ =~ z1 + z2 + z3
fY =~ y1 + y2 + y3
fXZ =~ x1z1R + x2z2R + x3z3R

fY ~ fX + fZ + fXZ

fX ~~ fZ
fX ~~ 0*fXZ
fZ ~~ 0*fXZ
"
out4 <-
    sem(mod4, data = dat2, std.lv = TRUE, meanstructure =
        TRUE)
summary(out4)
```

Example

```
lavaan (0.5-20) converged normally after  29 iterations

  Number of observations                           500

  Estimator                                         ML
  Minimum Function Test Statistic               45.602
  Degrees of freedom                                50
  P-value (Chi-square)                           0.650

Parameter Estimates:

  Information                                 Expected
  Standard Errors                             Standard

Latent Variables:
                   Estimate  Std.Err  Z-value  P(>|z|)
  fX =~
    x1                0.670    0.043   15.422    0.000
    x2                0.660    0.043   15.256    0.000
    x3                0.703    0.045   15.565    0.000
  fZ =~
    z1                0.738    0.048   15.342    0.000
```

Example

```
    z2                0.734    0.048   15.156    0.000
    z3                0.718    0.046   15.602    0.000
  fY =~
    y1                0.391    0.047    8.263    0.000
    y2                0.365    0.045    8.170    0.000
    y3                0.379    0.046    8.273    0.000
  fXZ =~
    x1z1R             0.365    0.057    6.378    0.000
    x2z2R             0.639    0.077    8.319    0.000
    x3z3R             0.501    0.066    7.563    0.000

Regressions:
                   Estimate  Std.Err  Z-value  P(>|z|)
  fY ~
    fX                1.676    0.248    6.756    0.000
    fZ               -0.075    0.100   -0.750    0.453
    fXZ               0.516    0.134    3.836    0.000

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

Example

```
  fZ ~~
    fXZ                0.000

Intercepts:
                   Estimate  Std.Err  Z-value  P(>|z|)
    x1               -0.011    0.044   -0.250    0.803
    x2               -0.033    0.044   -0.762    0.446
    x3               -0.027    0.046   -0.594    0.552
    z1                0.035    0.046    0.765    0.444
    z2                0.040    0.047    0.868    0.386
    z3                0.028    0.044    0.640    0.522
    y1                0.022    0.047    0.461    0.645
    y2                0.055    0.046    1.192    0.233
    y3                0.067    0.045    1.484    0.138
    x1z1R            -0.000    0.042   -0.000    1.000
    x2z2R            -0.000    0.046   -0.000    1.000
    x3z3R             0.000    0.044    0.000    1.000
    fX                0.000
    fZ                0.000
    fY                0.000
    fXZ               0.000

Variances:
```

Example

	Estimate	Std.Err	Z-value	P(>|z|)
x1	0.511	0.042	12.089	0.000
x2	0.514	0.042	12.217	0.000
x3	0.548	0.046	11.975	0.000
z1	0.523	0.052	10.142	0.000
z2	0.546	0.052	10.444	0.000
z3	0.461	0.048	9.704	0.000
y1	0.495	0.043	11.379	0.000
y2	0.541	0.044	12.324	0.000
y3	0.445	0.040	11.187	0.000
x1z1R	0.728	0.055	13.249	0.000
x2z2R	0.673	0.093	7.268	0.000
x3z3R	0.729	0.069	10.499	0.000
fX	1.000			
fZ	1.000			
fY	1.000			
fXZ	1.000			

Example

```
round(fitMeasures(out4)[c("chisq", "df", "pvalue", "cfi",
                          "tli", "rmsea", "srmr")], 3)
```

```
 chisq      df  pvalue     cfi     tli   rmsea    srmr
45.602  50.000   0.650   1.000   1.004   0.000   0.019
```

```
fitMeasures(out3)[c("aic", "bic")]
```

```
     aic      bic
22134.09 22378.54
```

```
fitMeasures(out4)[c("aic", "bic")]
```

```
     aic      bic
15754.44 15923.02
```

Example

```r
probeOut4 <- probe2WayRC(fit = out4,
                         nameX = c("fX", "fZ", "fXZ"),
                         nameY = "fY",
                         modVar = "fZ",
                         valProbe = c(-1, 0, 1)
                         )
probeOut4$SimpleSlope
```

```
     fZ    Slope         SE      Wald p
[1,] -1 1.160686 0.16865988  6.881816 0
[2,]  0 1.676276 0.09909026 16.916660 0
[3,]  1 2.191866 0.15282623 14.342213 0
```

Double Mean Centering

Using the same problem setup as above, we could perform double mean centering by:

1. Mean center every indicator of X and Z:

$$
\begin{aligned}
x_1^c &= x_1 - \bar{x}_1 \\
&\vdots \\
z_1^c &= z_1 - \bar{z}_1 \\
&\vdots
\end{aligned}
$$

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

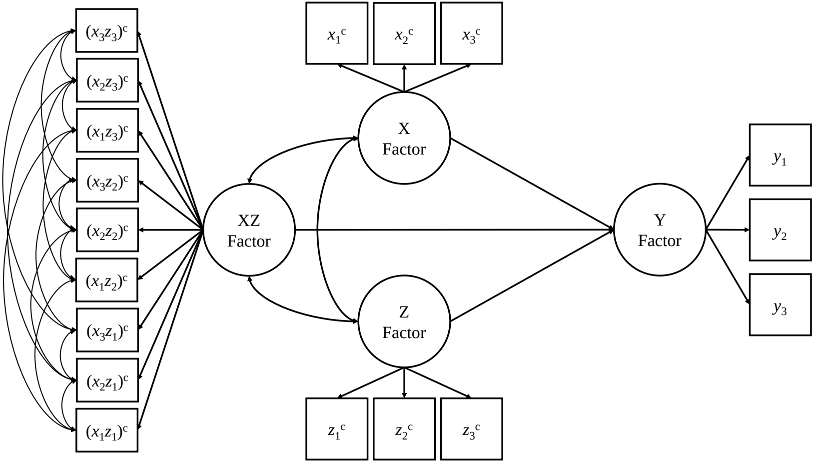
Double Mean Centering

3. Mean center each product term:

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

4. Use the mean centered indicators of X and Z, and the "double mean centered" product terms to specify the latent interaction model as represented in the following figure.

Double Mean Centering



Example

```
dat3 ← data.frame(lapply(dat1, scale, scale = FALSE))
tmpDat ← data.frame(
    x1z1 = with(dat3, x1*z1),
    x1z2 = with(dat3, x1*z2),
    x1z3 = with(dat3, x1*z3),

    x2z1 = with(dat3, x2*z1),
    x2z2 = with(dat3, x2*z2),
    x2z3 = with(dat3, x2*z3),

    x3z1 = with(dat3, x3*z1),
    x3z2 = with(dat3, x3*z2),
    x3z3 = with(dat3, x3*z3)
)
dat3 ← data.frame(dat3,
                  lapply(tmpDat, scale, scale = FALSE)
                  )
```

Example

```
mod5 <- "
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 ~~ fZ

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

```
"
out5 ← sem(mod5, data = dat3, std.lv = TRUE)
summary(out5)
```

```
lavaan (0.5-20) converged normally after  51 iterations

  Number of observations                           500

  Estimator                                         ML
  Minimum Function Test Statistic              134.186
  Degrees of freedom                               111
  P-value (Chi-square)                           0.066

Parameter Estimates:

  Information                                 Expected
  Standard Errors                             Standard

Latent Variables:
                   Estimate  Std.Err  Z-value  P(>|z|)
  fX =~
    x1                0.673    0.043   15.555    0.000
```

Example

```
    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
  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

Regressions:
                 Estimate  Std.Err  Z-value  P(>|z|)
```

Example

```
  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
  x1z1 ~~
    x1z2              0.274    0.033    8.339    0.000
    x1z3              0.309    0.033    9.239    0.000
    x2z1              0.240    0.031    7.675    0.000
    x3z1              0.240    0.032    7.471    0.000
  x1z2 ~~
    x1z3              0.232    0.032    7.217    0.000
    x2z2              0.218    0.036    6.031    0.000
    x3z2              0.250    0.041    6.128    0.000
  x1z3 ~~
    x2z3              0.214    0.031    6.939    0.000
    x3z3              0.211    0.034    6.115    0.000
  x2z1 ~~
    x2z2              0.245    0.042    5.794    0.000
```

Example

```
    x2z3              0.257     0.039     6.530     0.000
    x3z1              0.242     0.033     7.310     0.000
  x2z2 ~~
    x2z3              0.307     0.043     7.181     0.000
    x3z2              0.202     0.040     5.007     0.000
  x2z3 ~~
    x3z3              0.137     0.031     4.403     0.000
  x3z1 ~~
    x3z2              0.218     0.041     5.271     0.000
    x3z3              0.260     0.040     6.416     0.000
  x3z2 ~~
    x3z3              0.156     0.041     3.787     0.000
  fX ~~
    fXZ              -0.087     0.067    -1.297     0.195
  fZ ~~
    fXZ               0.040     0.066     0.613     0.540

Variances:
                   Estimate  Std.Err  Z-value  P(>|z|)
    x1                0.507     0.042    12.086     0.000
    x2                0.516     0.042    12.309     0.000
    x3                0.550     0.046    12.075     0.000
    z1                0.522     0.052    10.122     0.000
```

Example

z2	0.547	0.052	10.457	0.000
z3	0.462	0.047	9.724	0.000
y1	0.495	0.043	11.391	0.000
y2	0.542	0.044	12.336	0.000
y3	0.444	0.040	11.188	0.000
x1z1	0.752	0.051	14.893	0.000
x1z2	0.757	0.056	13.619	0.000
x1z3	0.698	0.051	13.693	0.000
x2z1	0.659	0.057	11.657	0.000
x2z2	0.726	0.068	10.733	0.000
x2z3	0.679	0.056	12.086	0.000
x3z1	0.751	0.060	12.594	0.000
x3z2	0.727	0.071	10.291	0.000
x3z3	0.705	0.061	11.585	0.000
fX	1.000			
fZ	1.000			
fY	1.000			
fXZ	1.000			

Example

```
round(fitMeasures(out5)[c("chisq", "df", "pvalue", "cfi",
                          "tli", "rmsea", "srmr")], 3)
```

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

```
out5.2 ←
    sem(mod5, data = dat3, std.lv = TRUE, meanstructure =
        TRUE)
probeOut5 ← probe2WayMC(fit = out5.2,
                        nameX = c("fX", "fZ", "fXZ"),
                        nameY = "fY",
                        modVar = "fZ",
                        valProbe = c(-1, 0, 1)
                        )
probeOut5$SimpleSlope
```

```
     fZ    Slope        SE      Wald p
[1,] -1 1.200206 0.1940921  6.183693 0
[2,]  0 1.757342 0.1049521 16.744230 0
[3,]  1 2.314478 0.1546000 14.970750 0
```

Example

```
mod6 <- "
fX =~ x1 + x2 + x3
fZ =~ z1 + z2 + z3
fY =~ y1 + y2 + y3
fXZ =~ x1z1 + x2z2 + x3z3

fY ~ fX + fZ + fXZ

fX ~~ fZ
"
out6 <-
    sem(mod6, data = dat3, std.lv = TRUE, meanstructure =
        TRUE)
summary(out6)
```

Example

```
lavaan (0.5-20) converged normally after  29 iterations

  Number of observations                           500

  Estimator                                         ML
  Minimum Function Test Statistic               61.353
  Degrees of freedom                                48
  P-value (Chi-square)                           0.093

Parameter Estimates:

  Information                                 Expected
  Standard Errors                             Standard

Latent Variables:
                   Estimate  Std.Err  Z-value  P(>|z|)
  fX =~
    x1                0.673    0.043   15.578    0.000
    x2                0.661    0.043   15.324    0.000
    x3                0.700    0.045   15.519    0.000
  fZ =~
    z1                0.739    0.048   15.386    0.000
```

Example

```
    z2                0.733    0.048   15.130    0.000
    z3                0.718    0.046   15.605    0.000
  fY =~
    y1                0.375    0.051    7.383    0.000
    y2                0.349    0.048    7.318    0.000
    y3                0.363    0.049    7.389    0.000
  fXZ =~
    x1z1              0.379    0.057    6.599    0.000
    x2z2              0.616    0.073    8.404    0.000
    x3z3              0.522    0.066    7.900    0.000

Regressions:
                   Estimate  Std.Err  Z-value  P(>|z|)
  fY ~
    fX                1.833    0.303    6.041    0.000
    fZ               -0.137    0.112   -1.222    0.222
    fXZ               0.629    0.169    3.714    0.000

Covariances:
                   Estimate  Std.Err  Z-value  P(>|z|)
  fX ~~
    fZ                0.231    0.058    3.984    0.000
    fXZ              -0.111    0.072   -1.544    0.123
```

Example

```
  fZ ~~
    fXZ               0.064    0.071    0.901    0.368

Intercepts:
                   Estimate  Std.Err  Z-value  P(>|z|)
    x1                0.000    0.044    0.000    1.000
    x2                0.000    0.044    0.000    1.000
    x3               -0.000    0.046   -0.000    1.000
    z1               -0.000    0.046   -0.000    1.000
    z2               -0.000    0.047   -0.000    1.000
    z3                0.000    0.044    0.000    1.000
    y1               -0.000    0.047   -0.000    1.000
    y2               -0.000    0.046   -0.000    1.000
    y3                0.000    0.045    0.000    1.000
    x1z1              0.000    0.042    0.000    1.000
    x2z2              0.000    0.047    0.000    1.000
    x3z3              0.000    0.045    0.000    1.000
    fX                0.000
    fZ                0.000
    fY                0.000
    fXZ               0.000

Variances:
```

Example

	Estimate	Std.Err	Z-value	P(>|z|)
x1	0.506	0.042	12.088	0.000
x2	0.513	0.042	12.281	0.000
x3	0.553	0.046	12.134	0.000
z1	0.520	0.052	10.089	0.000
z2	0.549	0.052	10.503	0.000
z3	0.461	0.047	9.722	0.000
y1	0.494	0.043	11.374	0.000
y2	0.542	0.044	12.339	0.000
y3	0.444	0.040	11.189	0.000
x1z1	0.729	0.056	13.102	0.000
x2z2	0.720	0.087	8.321	0.000
x3z3	0.719	0.070	10.215	0.000
fX	1.000			
fZ	1.000			
fY	1.000			
fXZ	1.000			

Example

```
round(fitMeasures(out6)[c("chisq", "df", "pvalue", "cfi",
                          "tli", "rmsea", "srmr")], 3)
```

```
 chisq      df  pvalue     cfi     tli   rmsea    srmr
61.353  48.000   0.093   0.991   0.987   0.024   0.026
```

```
fitMeasures(out5)[c("aic", "bic")]
```

```
     aic       bic
22197.38  22450.25
```

```
fitMeasures(out6)[c("aic", "bic")]
```

```
     aic       bic
15774.19  15951.20
```

Example

```
probeOut6 <- probe2WayMC(fit = out6,
                         nameX = c("fX", "fZ", "fXZ"),
                         nameY = "fY",
                         modVar = "fZ",
                         valProbe = c(-1, 0, 1)
                         )
probeOut6$SimpleSlope
```

```
     fZ    Slope        SE      Wald p
[1,] -1 1.204548 0.2303323  5.229609 0
[2,]  0 1.833131 0.1120757 16.356193 0
[3,]  1 2.461715 0.1713437 14.367117 0
```

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

```r
## Use semTools to orthogonalize:
dat2.2 ← indProd(data = dat1,
                 var1 = c("x1", "x2", "x3"),
                 var2 = c("z1", "z2", "z3"),
                 match = FALSE,
                 residualC = TRUE)
sum(dat2 - dat2.2)
```

```
[1] -9.790839e-14
```

```r
##
## Use semTools to double mean center:
dat3.2 ← indProd(data = dat1,
                 var1 = c("x1", "x2", "x3"),
                 var2 = c("z1", "z2", "z3"),
                 match = FALSE,
                 doubleMC = TRUE)
sum(dat3[ , -c(1 : 9)] - dat3.2[ , -c(1 : 9)])
```

```
[1] 0
```

Other Things

Moderation in logistic regression:

- Nothing special
- Just include the product term as a predictor
- Make sure to keep track of the weird "multiplicative change in log-odds" interpretation of your coefficients

Other Things

Moderation in logistic regression:

- Nothing special
- Just include the product term as a predictor
- Make sure to keep track of the weird "multiplicative change in log-odds" interpretation of your coefficients

Effect size for conditional process analysis:

- We don't know
- I could not find any work directly addressing the issue
- Fully and partially standardized indirect effects seem like they should still work
- κ^2 and the various flavors of R^2 aren't so clear-cut.

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.