# Structural Equation Modeling & Mediation

## Introduction to SEM with Lavaan

Kyle M. Lang

Department of Methodology & Statistics
Utrecht University

# Outline

Structural Equation Modeling

Mediation

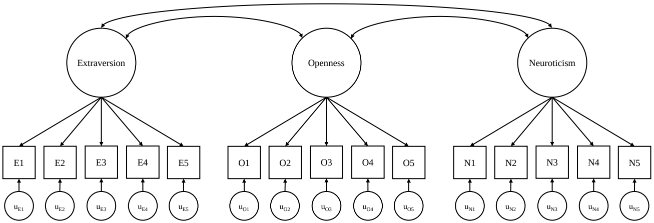
- Simple Mediation
- Bootstrapping
- Multiple Mediation
  - Parallel Mediators
  - Serial Mediators
- Mediation + SEM

# Full SEM

Structural equation modeling (SEM) simply combines path analysis and CFA.

- SEM allows us to model complicated structural relations among latent variables.

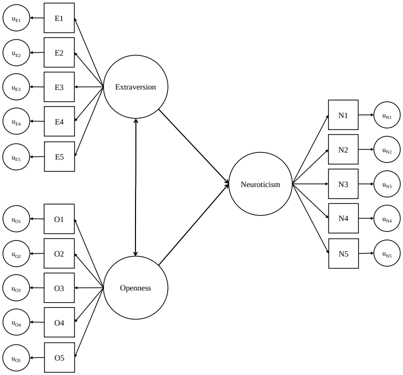
Let's consider a simple, three-factor CFA model.



# CFA → SEM

We first evaluate the validity of the measurement model via CFA.

- We then convert the CFA to an SEM by converting some covariances to latent regression paths.



# CFA Example

```
data(bfi, package = "psych")

## Specify the CFA model:
cfaMod <- "
extra =~ E1 + E2 + E3 + E4 + E5
open  =~ O1 + O2 + O3 + O4 + O5
neuro =~ N1 + N2 + N3 + N4 + N5
"

## Estimate the model:
cfaOut <- cfa(cfaMod, data = bfi, missing = "fiml", std.lv = TRUE)

## Check the fit:
fitMeasures(cfaOut,
            c("chisq", "df", "pvalue", "cfi", "tli", "rmsea", "srmr")
            )

   chisq       df   pvalue      cfi      tli    rmsea
2251.679   87.000    0.000    0.809    0.769    0.094
    srmr
   0.081
```

# CFA Example

```
partSummary(cfaOut, 5)

Latent Variables:
                   Estimate  Std.Err  z-value  P(>|z|)
  extra =~
    E1                0.973    0.032   30.607    0.000
    E2                1.163    0.030   38.171    0.000
    E3               -0.815    0.027  -30.358    0.000
    E4               -0.979    0.028  -35.254    0.000
    E5               -0.714    0.027  -26.638    0.000
  open =~
    O1                0.630    0.025   24.886    0.000
    O2               -0.605    0.036  -16.781    0.000
    O3                0.897    0.029   30.765    0.000
    O4                0.290    0.028   10.402    0.000
    O5               -0.602    0.031  -19.734    0.000
  neuro =~
    N1                1.272    0.027   47.254    0.000
    N2                1.218    0.026   46.491    0.000
    N3                1.157    0.029   40.195    0.000
    N4                0.892    0.030   29.356    0.000
    N5                0.823    0.031   26.163    0.000
```

# CFA Example

```
partSummary(cfaOut, 6)

Covariances:
                   Estimate  Std.Err  z-value  P(>|z|)
  extra ~~
    open             -0.444    0.024  -18.472    0.000
    neuro             0.240    0.023   10.551    0.000
  open ~~
    neuro            -0.117    0.025   -4.667    0.000
```

# CFA Example

```
partSummary(cfaOut, 7)

Intercepts:
                   Estimate  Std.Err  z-value  P(>|z|)
   .E1                2.974    0.031   96.223    0.000
   .E2                3.143    0.030  103.424    0.000
   .E3                4.002    0.026  156.117    0.000
   .E4                4.421    0.028  160.350    0.000
   .E5                4.417    0.025  174.595    0.000
   .O1                4.816    0.021  224.964    0.000
   .O2                2.713    0.030   91.745    0.000
   .O3                4.436    0.023  191.555    0.000
   .O4                4.892    0.023  211.519    0.000
   .O5                2.490    0.025   98.932    0.000
   .N1                2.932    0.030   98.589    0.000
   .N2                3.508    0.029  121.459    0.000
   .N3                3.217    0.030  106.147    0.000
   .N4                3.185    0.030  106.894    0.000
   .N5                2.969    0.031   96.663    0.000
    extra             0.000
    open              0.000
    neuro             0.000
```

# CFA Example

```
partSummary(cfaOut, 8)

Variances:
                   Estimate  Std.Err  z-value  P(>|z|)
   .E1                1.713    0.054   31.442    0.000
   .E2                1.224    0.049   24.952    0.000
   .E3                1.163    0.038   30.388    0.000
   .E4                1.166    0.041   28.522    0.000
   .E5                1.272    0.039   32.789    0.000
   .O1                0.878    0.031   28.320    0.000
   .O2                2.083    0.062   33.705    0.000
   .O3                0.686    0.043   16.130    0.000
   .O4                1.407    0.039   36.236    0.000
   .O5                1.401    0.044   31.837    0.000
   .N1                0.848    0.037   23.029    0.000
   .N2                0.842    0.035   24.184    0.000
   .N3                1.228    0.043   28.308    0.000
   .N4                1.666    0.051   32.808    0.000
   .N5                1.942    0.056   34.465    0.000
    extra             1.000
    open              1.000
    neuro             1.000
```

# SEM Example

```
## Add structural paths:
semMod <- "
extra =~ E1 + E2 + E3 + E4 + E5
open  =~ O1 + O2 + O3 + O4 + O5
neuro =~ N1 + N2 + N3 + N4 + N5

neuro ~ extra + open
"

## Estimate the model:
semOut <- sem(semMod, data = bfi, missing = "fiml", std.lv = TRUE)

## Check the fit:
fitMeasures(semOut,
            c("chisq", "df", "pvalue", "cfi", "tli", "rmsea", "srmr")
            )

   chisq       df   pvalue      cfi      tli    rmsea
2251.679   87.000    0.000    0.809    0.769    0.094
    srmr
   0.081
```

# SEM Example

```
partSummary(semOut, 5)

Latent Variables:
                   Estimate  Std.Err  z-value  P(>|z|)
  extra =~
    E1                0.973    0.032   30.607    0.000
    E2                1.163    0.030   38.172    0.000
    E3               -0.815    0.027  -30.358    0.000
    E4               -0.979    0.028  -35.254    0.000
    E5               -0.714    0.027  -26.638    0.000
  open =~
    O1                0.630    0.025   24.886    0.000
    O2               -0.605    0.036  -16.781    0.000
    O3                0.897    0.029   30.765    0.000
    O4                0.290    0.028   10.402    0.000
    O5               -0.602    0.031  -19.734    0.000
  neuro =~
    N1                1.235    0.027   45.916    0.000
    N2                1.183    0.026   45.360    0.000
    N3                1.123    0.028   39.976    0.000
    N4                0.866    0.029   29.605    0.000
    N5                0.799    0.031   26.204    0.000
```

# SEM Example

```
partSummary(semOut, 6:7)

Regressions:
                   Estimate  Std.Err  z-value  P(>|z|)
  neuro ~
    extra             0.241    0.030    8.169    0.000
    open             -0.014    0.031   -0.448    0.654

Covariances:
                   Estimate  Std.Err  z-value  P(>|z|)
  extra ~~
    open             -0.444    0.024  -18.472    0.000
```

# SEM Example

```
partSummary(semOut, 8)

Intercepts:
                   Estimate  Std.Err  z-value  P(>|z|)
   .E1                2.974    0.031   96.223    0.000
   .E2                3.143    0.030  103.424    0.000
   .E3                4.002    0.026  156.117    0.000
   .E4                4.421    0.028  160.350    0.000
   .E5                4.417    0.025  174.595    0.000
   .O1                4.816    0.021  224.964    0.000
   .O2                2.713    0.030   91.745    0.000
   .O3                4.436    0.023  191.555    0.000
   .O4                4.892    0.023  211.520    0.000
   .O5                2.490    0.025   98.932    0.000
   .N1                2.932    0.030   98.589    0.000
   .N2                3.508    0.029  121.459    0.000
   .N3                3.217    0.030  106.146    0.000
   .N4                3.185    0.030  106.894    0.000
   .N5                2.969    0.031   96.663    0.000
    extra             0.000
    open              0.000
   .neuro             0.000
```

# SEM Example

```
partSummary(semOut, 9)

Variances:
                   Estimate  Std.Err  z-value  P(>|z|)
   .E1                1.713    0.054   31.442    0.000
   .E2                1.224    0.049   24.952    0.000
   .E3                1.163    0.038   30.388    0.000
   .E4                1.166    0.041   28.522    0.000
   .E5                1.272    0.039   32.789    0.000
   .O1                0.878    0.031   28.320    0.000
   .O2                2.083    0.062   33.705    0.000
   .O3                0.686    0.043   16.130    0.000
   .O4                1.407    0.039   36.236    0.000
   .O5                1.401    0.044   31.837    0.000
   .N1                0.848    0.037   23.029    0.000
   .N2                0.842    0.035   24.184    0.000
   .N3                1.228    0.043   28.308    0.000
   .N4                1.666    0.051   32.808    0.000
   .N5                1.942    0.056   34.465    0.000
    extra             1.000
    open              1.000
   .neuro             1.000
```

# Why SEM?

The beauty of SEM is that we get to model the types of complex relations we can specify via path models while leveraging all the strengths of latent variables.

- When we fit a multiple-group SEM, we are modeling moderation by group.
    - The latent variables give us the ability to evaluate measurement invariance across groups.
    - We'll see more of these ideas in the next lecture.
- Path analysis and SEM lend themselves especially well to mediation analysis and conditional process anlaysis.

# MEDIATION

# Mediation vs. Moderation

What do we mean by *mediation* and *moderation*?

Mediation and moderation are types of hypotheses, not statistical methods or models.

- Mediation tells us *how* one variable influences another.
- Moderation tells us *when* one variable influences another.

# Contextualizing Example

Say we wish to explore the process underlying exercise habits.

Our first task is to operationalize “exercise habits”

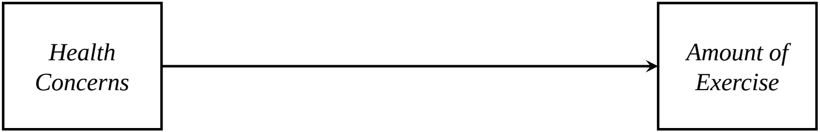
- DV: Hours per week spent in vigorous exercise (*exerciseAmount*).

We may initial ask: what predicts devoting more time to exercise?

- IV: Concerns about negative health outcomes (*healthConcerns*).

# Focal Effect Only

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



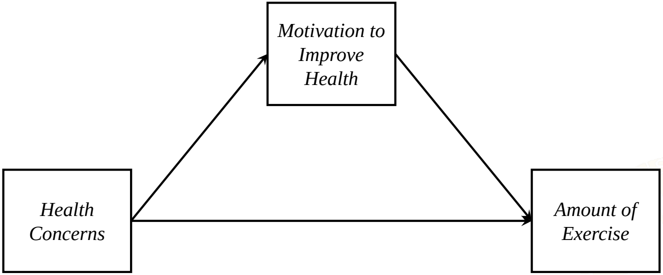

- Mediation, moderation, and conditional process analysis all attempt to describe the focal effect in more detail.
- We always begin by hypothesizing a focal effect.

# The 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*).

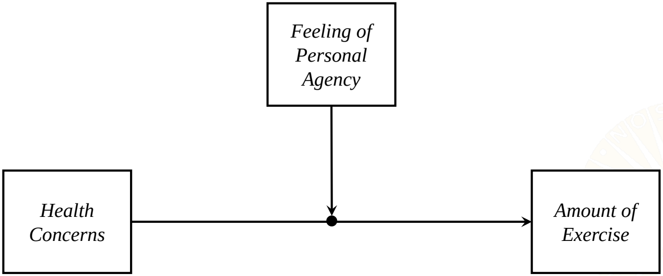


# Moderation Hypothesis

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*).

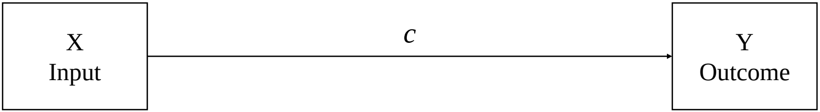


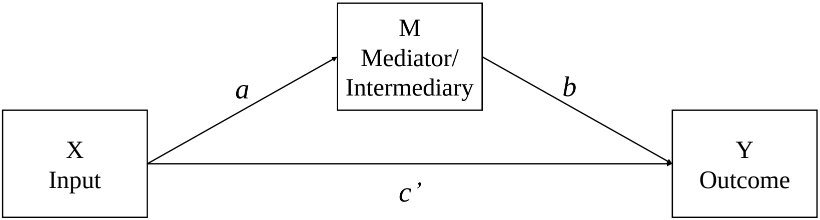
# Conditional Process Analysis

Conditional process analysis combines the mediation and moderation hypotheses into models of moderated mediation.

- Given a mediation model describing *how* health concerns affect exercise amount, what other variables may modulate the indirect effect.

# Path Diagrams



# Necessary Equations

To get all the pieces of the preceding diagram using OLS regression, we'll need to fit three seperate models.

$$Y = i_1 + cX + e_1 \tag{1}$$
$$Y = i_2 + c'X + bM + e_2 \tag{2}$$
$$M = i_3 + aX + e_3 \tag{3}$$

- Equation 1 gives us the total effect ($c$).
- Equation 2 gives us the direct effect ($c'$) and the partialled effect of the mediator on the outcome ($b$).
- Equation 3 gives us the effect of the input on the outcome ($a$).

# Two Measures of Indirect Effect

Indirect effects can be quantified in two different ways:

$$IE_{diff} = c - c' \tag{4}$$

$$IE_{prod} = a \cdot b \tag{5}$$

$IE_{diff}$ and $IE_{prod}$ are equivalent in simple mediation.

- Both give us information about the proportion of the total effect that is transmitted through the intermediary variable.
- $IE_{prod}$ provides a more direct representation of the actual pathway we're interested in testing.
- $IE_{diff}$ gets at our desired hypothesis indirectly.

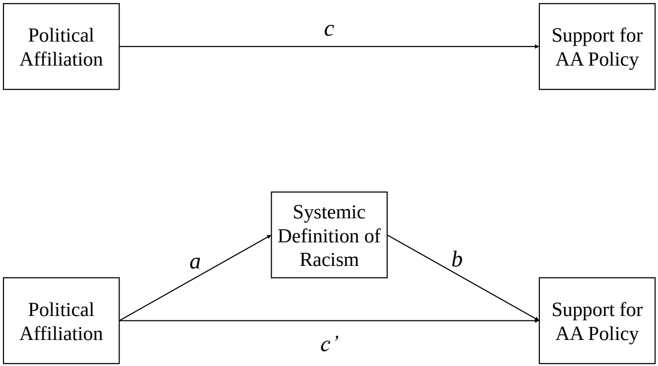
# The Causal Steps Approach

Baron and Kenny (1986, p. 1176) describe three/four conditions as being sufficient to demonstrate statistical "mediation."

1. Variations in levels of the independent variable significantly account for variations in the presumed mediator (i.e., Path *a*).
    - Need a significant $a$ path.
2. Variations in the mediator significantly account for variations in the dependent variable (i.e., Path *b*).
    - Need a significant $b$ path.
3. When Paths *a* and *b* are controlled, a previously significant relation between the independent and dependent variables is no longer significant.
    - Need a significant total effect
    - The direct effect must be "less" than the total effect

# Example Process Model

Consider the following process.



# Causal Steps Example

```r
## Load some data:
dat1 <- readRDS("../data/adamsKlpsScaleScore.rds")

## Check pre-conditions:
mod1 <- lm(policy ~ polAffil, data = dat1)
mod2 <- lm(policy ~ sysRac, data = dat1)
mod3 <- lm(sysRac ~ polAffil, data = dat1)

## Partial out the mediator's effect:
mod4 <- lm(policy ~ sysRac + polAffil, data = dat1)
```

# Causal Steps Example

```
summary(mod1)


Call:
lm(formula = policy ~ polAffil, data = dat1)

Residuals:
    Min      1Q  Median      3Q     Max 
-2.7357 -0.8254  0.0643  0.6827  3.2481 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  2.71516    0.35648   7.617 3.32e-11 ***
polAffil     0.23675    0.07775   3.045   0.0031 ** 
---
Signif. codes:  
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.134 on 85 degrees of freedom
Multiple R-squared:  0.09836,Adjusted R-squared:  0.08775 
F-statistic: 9.273 on 1 and 85 DF,  p-value: 0.003096
```

# Causal Steps Example

```
summary(mod2)


Call:
lm(formula = policy ~ sysRac, data = dat1)

Residuals:
     Min       1Q   Median       3Q      Max
-2.28970 -0.53821  0.08866  0.64015  3.08343

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   1.1218     0.4883   2.297   0.0241 *
sysRac        0.6649     0.1210   5.494 4.03e-07 ***
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.026 on 85 degrees of freedom
Multiple R-squared:  0.262,Adjusted R-squared:  0.2534
F-statistic: 30.18 on 1 and 85 DF,  p-value: 4.029e-07
```

# Causal Steps Example

```
summary(mod3)


Call:
lm(formula = sysRac ~ polAffil, data = dat1)

Residuals:
    Min      1Q  Median      3Q     Max 
-2.2187 -0.5449 -0.2115  0.6182  1.9516 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  3.19726    0.27634  11.570   <2e-16 ***
polAffil     0.17023    0.06027   2.825   0.0059 ** 
---
Signif. codes:  
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.8788 on 85 degrees of freedom
Multiple R-squared:  0.08581,Adjusted R-squared:  0.07505 
F-statistic: 7.978 on 1 and 85 DF,  p-value: 0.005898
```

# Causal Steps Example

```
summary(mod4)


Call:
lm(formula = policy ~ sysRac + polAffil, data = dat1)

Residuals:
    Min      1Q  Median      3Q     Max
-2.1370 -0.6338 -0.0020  0.6658  3.4674

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.80704    0.51013   1.582   0.1174
sysRac       0.59680    0.12478   4.783  7.3e-06 ***
polAffil     0.13515    0.07252   1.864   0.0658 .
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.011 on 84 degrees of freedom
Multiple R-squared:  0.2913,Adjusted R-squared:  0.2745
F-statistic: 17.27 on 2 and 84 DF,  p-value: 5.228e-07
```

# Causal Steps Example

```
## Extract important parameter estimates:
a      <- coef(mod3)["polAffil"]
b      <- coef(mod4)["sysRac"]
c      <- coef(mod1)["polAffil"]
cPrime <- coef(mod4)["polAffil"]

## Compute indirect effects:
ieDiff <- unname(c - cPrime)
ieProd <- unname(a * b)

ieDiff

[1] 0.1015958

ieProd

[1] 0.1015958
```

# Sobel's Z

In the previous example, do we have a *significant* indirect effect?

- The direct effect is “substantially” smaller than the total effect, but is the difference statistically significant?
- Sobel (1982) developed an asymptotic standard error for $IE_{prod}$ that we can use to assess this hypothesis.

$$SE_{sobel} = \sqrt{a^2 \cdot SE_b^2 + b^2 \cdot SE_a^2} \quad (6)$$

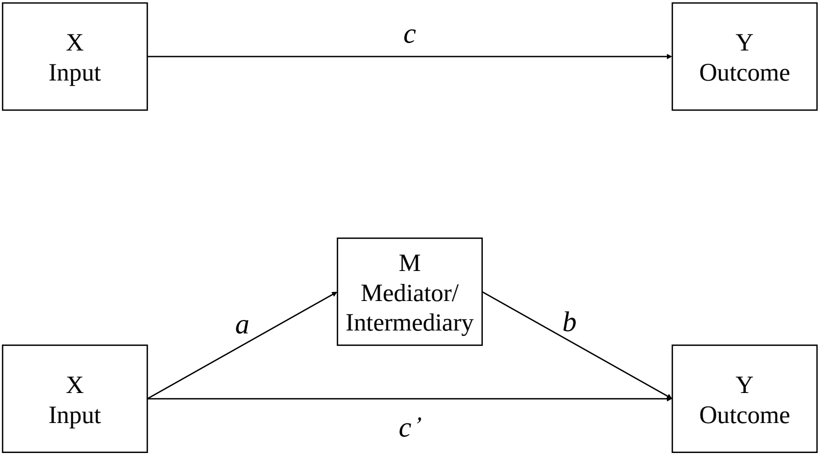
$$Z_{sobel} = \frac{ab}{SE_{sobel}} \quad (7)$$

$$95\%CI_{sobel} = ab \pm 1.96 \cdot SE_{sobel} \quad (8)$$

# Sobel Example

```
## SE:
seA <- (mod3 %>% vcov() %>% diag() %>% sqrt())["polAffil"]
seB <- (mod4 %>% vcov() %>% diag() %>% sqrt())["sysRac"]

se <- sqrt(b^2 * seA^2 + a^2 * seB^2) %>% unname()

## z-score:
(z <- ieProd / se)

[1] 2.432107

## p-value:
(p <- 2 * pnorm(z, lower = FALSE))

[1] 0.01501126

## 95% CI:
c(ieProd - 1.96 * se, ieProd + 1.96 * se)

[1] 0.01972121 0.18347034
```

# Recall our Basic Path Diagram



# Two Measures of Indirect Effect

Recall the two definitions of an indirect effect:

$$IE_{diff} = c - c' \tag{9}$$

$$IE_{prod} = a \cdot b \tag{10}$$

It pays to remember a few key points:

- $IE_{diff}$ and $IE_{prod}$ are equivalent in simple mediation.
- $IE_{diff}$ is only an indirect indication of $IE_{prod}$.
- A significant indirect effect can exist without a significant total effect.
- If we only care about the indirect effect, then we don't need to worry about the total effect.

# Two Measures of Indirect Effect

Recall the two definitions of an indirect effect:

$$IE_{diff} = c - c' \tag{9}$$

$$IE_{prod} = a \cdot b \tag{10}$$

It pays to remember a few key points:

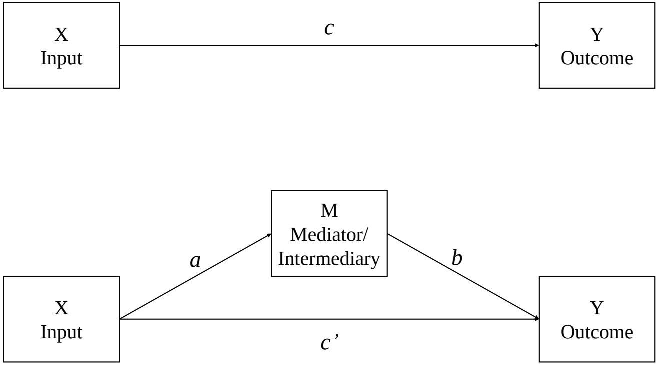
- $IE_{diff}$ and $IE_{prod}$ are equivalent in simple mediation.
- $IE_{diff}$ is only an indirect indication of $IE_{prod}$.
- A significant indirect effect can exist without a significant total effect.
- If we only care about the indirect effect, then we don't need to worry about the total effect.

These points imply something interesting:

- We don't need to estimate $c$!

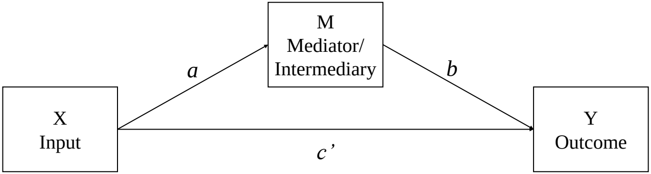
# Simplifying our Path Diagram

QUESTION: If we don't care about directly estimating $c$, how can we simplify this diagram?

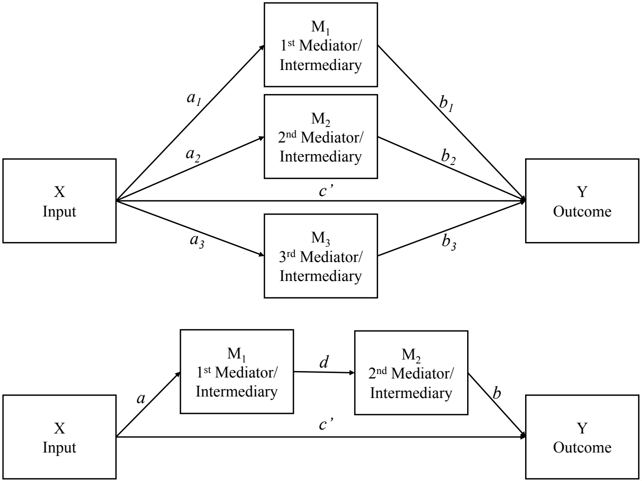


# Simplifying our Path Diagram

ANSWER: We don't fit the upper model.

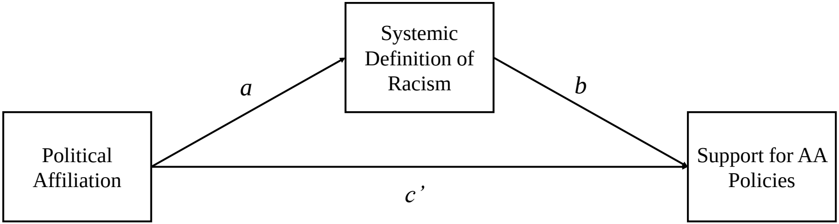


# Why Path Analysis?



# Example

Let's revisit the above example using path analysis in **lavaan**.



# Example

```r
## Load the lavaan package:
library(lavaan)

## Specify the basic path model:
mod1 <- "
policy ~ sysRac + polAffil
sysRac ~ polAffil
"

## Estimate the model:
out1 <- sem(mod1, data = dat1)
```

# Example

```
## Look at the results:
partSummary(out1, c(5, 6))

Regressions:
                   Estimate  Std.Err  z-value  P(>|z|)
  policy ~
    sysRac            0.597    0.123    4.867    0.000
    polAffil          0.135    0.071    1.897    0.058
  sysRac ~
    polAffil          0.170    0.060    2.858    0.004

Variances:
                   Estimate  Std.Err  z-value  P(>|z|)
   .policy            0.987    0.150    6.595    0.000
   .sysRac            0.755    0.114    6.595    0.000
```

# Example

```r
## Include the indirect effect:
mod2 <- "
policy ~ b*sysRac + polAffil
sysRac ~ a*polAffil

ab := a*b # Define a parameter for the indirect effect
"

## Estimate the model:
out2 <- sem(mod2, data = dat1)
```

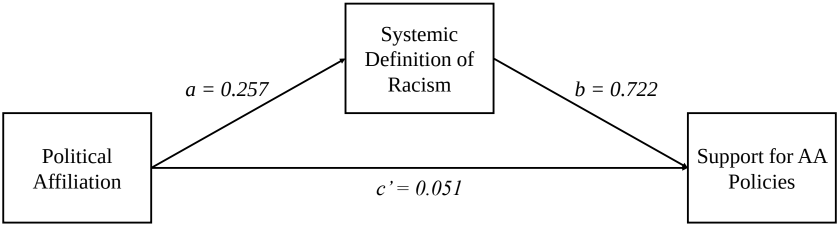
# Example

```
## Look at the results:
partSummary(out2, 5:7)

Regressions:
                   Estimate  Std.Err  z-value  P(>|z|)
  policy ~
    sysRac     (b)    0.597    0.123    4.867    0.000
    polAffil          0.135    0.071    1.897    0.058
  sysRac ~
    polAffil   (a)    0.170    0.060    2.858    0.004

Variances:
                   Estimate  Std.Err  z-value  P(>|z|)
   .policy            0.987    0.150    6.595    0.000
   .sysRac            0.755    0.114    6.595    0.000

Defined Parameters:
                   Estimate  Std.Err  z-value  P(>|z|)
    ab                0.102    0.041    2.464    0.014
```

# Example

```
## We can also get CIs:
parameterEstimates(out2, zstat = FALSE, pvalue = FALSE, ci = TRUE)

       lhs op      rhs label   est    se ci.lower ci.upper
1   policy  ~   sysRac     b 0.597 0.123    0.356    0.837
2   policy  ~ polAffil       0.135 0.071   -0.005    0.275
3   sysRac  ~ polAffil     a 0.170 0.060    0.053    0.287
4   policy ~~   policy       0.987 0.150    0.694    1.280
5   sysRac ~~   sysRac       0.755 0.114    0.530    0.979
6 polAffil ~~ polAffil       2.444 0.000    2.444    2.444
7       ab :=      a*b    ab 0.102 0.041    0.021    0.182
```

# Results


Systemic Definition of Racism
a = 0.257
b = 0.722
Political Affiliation
Support for AA Policies
c' = 0.051

# We’re not there yet...

Path analysis allows us to directly model complex (and simple) relations, but the preceding example still suffers from a considerable limitation.

- The significance test for the indirect effect is still conducted with the Sobel Z approach.

Path analysis (or full SEM) doesn’t magically get around distributional problems associated with Sobel’s Z test.

- To get a robust significance test of the indirect effect, we need to use *bootstrapping*.

# Bootstrapping

Bootstrapping was introduced by Efron (1979) as a tool for non-parametric inference.

- Traditional inference requires that we assume a parametric sampling distribution for our focal parameter.
- We need to make such an assumption to compute the standard errors we require for inferences.
- If we cannot safely make these assumptions, we can use bootstrapping.

# Bootstrapping

Assume our observed data $Data_0$ represent the population and:

1. Sample rows of $Data_0$, with replacement, to create $B$ new samples $\{Data_b\}$.
2. Calculate our focal statistic on each of the $B$ bootstrap samples.
3. Make inferences based on the empirical distribution of the $B$ estimates calculated in Step 2

# Bootstrapping

$\text{Data}_0 =$

| ID | $X$ | $Y$ |
|---|---|---|
| 1 | $x_{i=1}$ | $y_{i=1}$ |
| 2 | $x_{i=2}$ | $y_{i=2}$ |
| 3 | $x_{i=3}$ | $y_{i=3}$ |
| 4 | $x_{i=4}$ | $y_{i=4}$ |
| 5 | $x_{i=5}$ | $y_{i=5}$ |

$\text{Data}_{b=1} =$

| ID | $X$ | $Y$ |
|---|---|---|
| 3 | $x_{i=3}$ | $y_{i=3}$ |
| 2 | $x_{i=2}$ | $y_{i=2}$ |
| 4 | $x_{i=4}$ | $y_{i=4}$ |
| 2 | $x_{i=2}$ | $y_{i=2}$ |
| 3 | $x_{i=3}$ | $y_{i=3}$ |

, $\text{Data}_{b=2} =$

| ID | $X$ | $Y$ |
|---|---|---|
| 1 | $x_{i=1}$ | $y_{i=1}$ |
| 3 | $x_{i=3}$ | $y_{i=3}$ |
| 3 | $x_{i=3}$ | $y_{i=3}$ |
| 4 | $x_{i=4}$ | $y_{i=4}$ |
| 5 | $x_{i=5}$ | $y_{i=5}$ |

, $\text{Data}_{b=3} =$

| ID | $X$ | $Y$ |
|---|---|---|
| 3 | $x_{i=3}$ | $y_{i=3}$ |
| 5 | $x_{i=5}$ | $y_{i=5}$ |
| 1 | $x_{i=1}$ | $y_{i=1}$ |
| 4 | $x_{i=4}$ | $y_{i=4}$ |
| 3 | $x_{i=3}$ | $y_{i=3}$ |

, $\cdots$ $\text{Data}_{b=B} =$

| ID | $X$ | $Y$ |
|---|---|---|
| 2 | $x_{i=2}$ | $y_{i=2}$ |
| 5 | $x_{i=5}$ | $y_{i=5}$ |
| 1 | $x_{i=1}$ | $y_{i=1}$ |
| 1 | $x_{i=1}$ | $y_{i=1}$ |
| 4 | $x_{i=4}$ | $y_{i=4}$ |

# Example

Suppose I'm on the lookout for a retirement location. Since I want to relax in my old-age, I'm concerned with ensuring a low probability of dragon attacks, so I have a few salient considerations:

- Shooting for a location with no dragons, whatsoever, is a fools errand (since dragons are, obviously, ubiquitous).
- I merely require a location that has at least two times as many dragon-free days as other kinds.

# Example

I've been watching several candidate locales over the course of my (long and illustrious) career, and I'm particularly hopeful about one quiet hamlet in the Patagonian highlands.

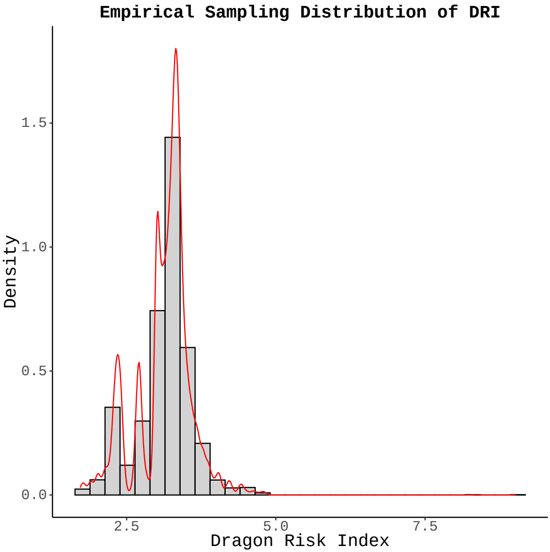
- To ensure that my required degree of dragon-freeness is met, I'll use the *Dragon Risk Index* (DRI):

$$DRI = Median\left(\frac{\text{Dragon-Free Days}}{\text{Dragonned Days}}\right)$$

# Example

```r
## Read in the observed data:
rawData <- readRDS("../data/daysData.rds")

## Compute the observed test statistic:
obsDRI <- median(rawData$goodDays / rawData$badDays)
obsDRI

[1] 3.24476

## Draw the bootstrap samples:
set.seed(235711)
nSams <- 5000
bootDRI <- rep(NA, nSams)
for(b in 1:nSams) {
    bootSam    <- rawData[sample(1:nrow(rawData), replace = TRUE), ]
    bootDRI[b] <- median(bootSam$goodDays / bootSam$badDays)
}
```

# Example



Empirical Sampling Distribution of DRI
Density
1.5
1.0
0.5
0.0
2.5
5.0
7.5
Dragon Risk Index

# Example

To see if I can be confident in the dragon-freeness of my potential home, I'll summarize the preceding distribution with a (one-tailed) percentile confidence interval:

```
bootLB <- sort(bootDRI)[0.05 * nSams]
bootUB <- Inf

## The bootstrapped Percentile CI:
c(bootLB, bootUB)

[1] 2.288555      Inf
```

# Bootstrapped Inference for Indirect Effects

We can apply the same procedure to testing the indirect effect.

- The problem with Sobel's Z is exactly the type of issue for which bootstrapping was designed
  - We don't know a reasonable finite-sample sampling distribution for the $ab$ parameter.
- Bootstrapping will allow us to construct an empirical sampling distribution for $ab$ and construct confidence intervals for inference.

# Bootstrapped Inference for Indirect Effects

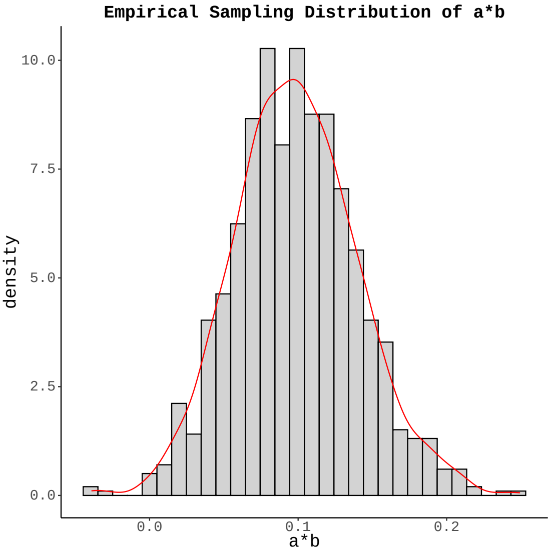
PROCEDURE:

1. Resample our observed data with replacement
2. Fit our hypothesized path model to each bootstrap sample
3. Store the value of $ab$ that we get each time
4. Summarize the empirical distribution of $ab$ to make inferences

# Example

```
nSams <- 1000
abVec <- rep(NA, nSams)
for(i in 1:nSams) {
    ## Resample the data:
    bootSam <- dat1[sample(1:nrow(dat1), replace = TRUE), ]

    ## Fit the path model:
    bootOut <- sem(mod2, data = bootSam)

    ## Store the estimated indirect effect:
    abVec[i] <- coef(bootOut)[c("a", "b")] %>% prod()
}
```

# Example



Empirical Sampling Distribution of a*b
density
10.0
7.5
5.0
2.5
0.0
0.0
0.1
0.2
a*b

# Example

```
## Calculate the percentile CI:
lb <- sort(abVec)[0.025 * nSams]
ub <- sort(abVec)[0.975 * nSams]
c(lb, ub)

[1] 0.01972696 0.18599868
```

# Example

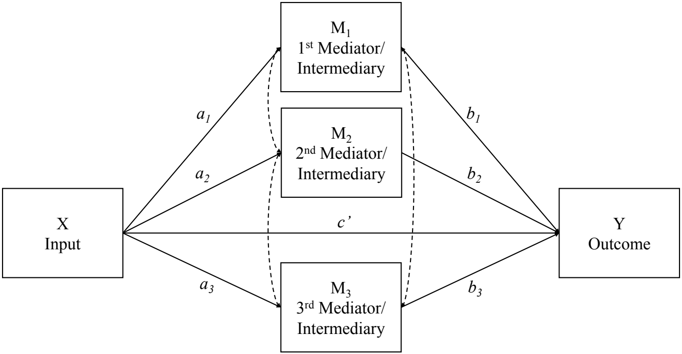
```r
## Much more parsimoniously:
bootOut2 <- sem(mod2, data = dat1, se = "boot", bootstrap = 1000)

parameterEstimates(bootOut2, zstat = FALSE, pvalue = FALSE)

       lhs op      rhs label   est    se ci.lower ci.upper
1   policy  ~   sysRac     b 0.597 0.134    0.331    0.850
2   policy  ~ polAffil       0.135 0.084   -0.019    0.310
3   sysRac  ~ polAffil     a 0.170 0.066    0.029    0.291
4   policy ~~   policy       0.987 0.161    0.661    1.288
5   sysRac ~~   sysRac       0.755 0.111    0.547    0.969
6 polAffil ~~ polAffil       2.444 0.000    2.444    2.444
7       ab :=      a*b    ab 0.102 0.042    0.016    0.184
```

# Simple Mediation is Too Simple

We can justify multiple mediator models by asking: “What mediates the effects in a simple mediation model?”

- Mediation of the direct effect leads to *parallel multiple mediator models*.
- Mediation of the $a$ or $b$ paths produces *serial multiple mediator models*.

# Parallel Multiple Mediation



# Parallel Multiple Mediation

To get all of the information in the preceding diagram, we need to estimate four equations:

$$
\begin{aligned}
Y &= i_Y + b_1 M_1 + b_2 M_2 + b_3 M_3 + c'X + e_Y \\
M_1 &= i_{M1} + a_1 X + e_{M1} \\
M_2 &= i_{M2} + a_2 X + e_{M2} \\
M_3 &= i_{M3} + a_3 X + e_{M3}
\end{aligned}
$$

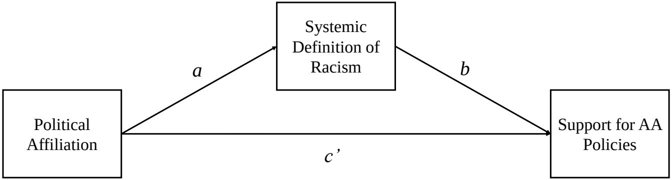
In general, a parallel mediator model with $K$ mediator variables will required $K + 1$ separate equations.

Path modeling can make this task much simpler.

- Also allows us to explicitly estimate the correlations between parallel mediators.

# Parallel Multiple Mediation

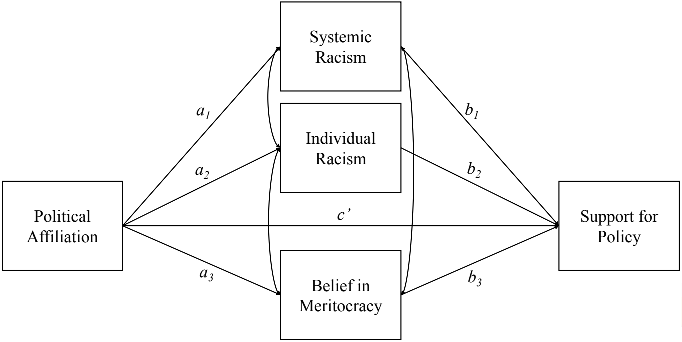
Let's reconsider the last example:




**QUESTION:** What might be mediating the residual direct effect?

# Parallel Multiple Mediation

## POTENTIAL ANSWER:



# A Quick Note on Inference

In parallel multiple mediation:

- We have $K$ *specific indirect effects*, where $K$ is the number of mediators: $a_1 b_1, a_2 b_2, \ldots, a_K b_K$.
- The *Total Indirect Effect* is equal to the sum of all the specific indirect effects: $IE_{tot} = \sum_{k=1}^{K} a_k b_k$.
- The *Total Effect* is equal to the direct effect plus the total indirect effect: $c = c' + IE_{tot}$

Inference for the specific indirect effects is basically the same as it is for the sole indirect effect in simple mediation.

- CAVEAT: Each specific indirect effect must be interpreted as conditional on all other mediators in the model.

# Example

```
## Read in the data
dat1 <- readRDS("../data/adamsKlpsScaleScore.rds")

## Parallel Multiple Mediator Model:
mod1.1 <- "
policy ~ b1*sysRac + b2*indRac + b3*merit + cp*polAffil
sysRac ~ a1*polAffil
indRac ~ a2*polAffil
merit ~ a3*polAffil

sysRac ~~ indRac + merit
indRac ~~ merit

ab1 := a1*b1
ab2 := a2*b2
ab3 := a3*b3
totalIE := ab1 + ab2 + ab3
"

## Fit the model:
out1.1 <- sem(mod1.1, data = dat1, se = "boot", bootstrap = 5000)
```

# Example

```
## Look at results:
partSummary(out1.1, 5)

Regressions:
                   Estimate  Std.Err  z-value  P(>|z|)
  policy ~
    sysRac    (b1)    0.601    0.142    4.238    0.000
    indRac    (b2)    0.143    0.106    1.353    0.176
    merit     (b3)   -0.036    0.152   -0.238    0.812
    polAffil  (cp)    0.125    0.077    1.639    0.101
  sysRac ~
    polAffil  (a1)    0.170    0.063    2.686    0.007
  indRac ~
    polAffil  (a2)   -0.004    0.079   -0.054    0.957
  merit ~
    polAffil  (a3)   -0.266    0.061   -4.352    0.000
```

# Example

```
partSummary(out1.1, 6:7)

Covariances:
                   Estimate  Std.Err  z-value  P(>|z|)
 .sysRac ~~
   .indRac           -0.076    0.100   -0.757    0.449
   .merit            -0.217    0.092   -2.354    0.019
 .indRac ~~
   .merit             0.154    0.098    1.570    0.116

Variances:
                   Estimate  Std.Err  z-value  P(>|z|)
   .policy            0.963    0.174    5.527    0.000
   .sysRac            0.755    0.110    6.864    0.000
   .indRac            1.188    0.156    7.609    0.000
   .merit             0.719    0.112    6.420    0.000
```

# Example

```
partSummary(out1.1, 8)

Defined Parameters:
                   Estimate  Std.Err  z-value  P(>|z|)
    ab1               0.102    0.044    2.310    0.021
    ab2              -0.001    0.014   -0.042    0.966
    ab3               0.010    0.042    0.231    0.817
    totalIE           0.111    0.051    2.165    0.030
```

# Example

```
parameterEstimates(out1.1, boot.ci.type = "bca.simple") %>%
    select(c("label", "est", "ci.lower", "ci.upper")) %>%
    tail(4)

     label    est ci.lower ci.upper
16     ab1  0.102    0.031    0.208
17     ab2 -0.001   -0.038    0.025
18     ab3  0.010   -0.080    0.087
19 totalIE  0.111    0.019    0.223
```

# Comparing Specific Indirect Effects

When we have multiple specific indirect effects in a single model, we can test if they are statistically different from one another.

QUESTION: How might we go about doing such a test (assuming we're using path modeling)?

# Comparing Specific Indirect Effects

When we have multiple specific indirect effects in a single model, we can test if they are statistically different from one another.

QUESTION: How might we go about doing such a test (assuming we're using path modeling)?

ANSWER: There are, at least, two reasonable methods:

1. Use nested model $\Delta\chi^2$ tests
2. Define a new parameter to encode the constraint and use bootstrapping

# Example

```
## Test differences in specific indirect effects:
mod1.2 <- "
policy ~ b1*sysRac + b2*indRac + b3*merit + cp*polAffil
sysRac ~ a1*polAffil
indRac ~ a2*polAffil
merit ~ a3*polAffil

sysRac ~~ indRac + merit
indRac ~~ merit

ab1 := a1*b1
ab2 := a2*b2
ab3 := a3*b3
totalIE := ab1 + ab2 + ab3

ab1 == ab2 # The first two IEs are constrained to equality
"

out1.2 <- sem(mod1.2, data = dat1)
```

# Example

```
## Look at results:
partSummary(out1.2, 5)

Regressions:
                   Estimate  Std.Err  z-value  P(>|z|)
  policy ~
    sysRac    (b1)    0.575    0.123    4.662    0.000
    indRac    (b2)    0.192    0.096    2.004    0.045
    merit     (b3)   -0.055    0.131   -0.416    0.678
    polAffil  (cp)    0.125    0.074    1.696    0.090
  sysRac ~
    polAffil  (a1)    0.027    0.025    1.082    0.279
  indRac ~
    polAffil  (a2)    0.082    0.067    1.222    0.222
  merit ~
    polAffil  (a3)   -0.217    0.055   -3.943    0.000
```

# Example

```
partSummary(out1.2, 6:7)

Covariances:
                   Estimate  Std.Err  z-value  P(>|z|)
 .sysRac ~~
   .indRac           -0.106    0.106   -0.995    0.320
   .merit            -0.234    0.086   -2.731    0.006
 .indRac ~~
   .merit             0.164    0.102    1.615    0.106

Variances:
                   Estimate  Std.Err  z-value  P(>|z|)
   .policy            0.967    0.147    6.595    0.000
   .sysRac            0.804    0.122    6.595    0.000
   .indRac            1.206    0.183    6.595    0.000
   .merit             0.724    0.110    6.595    0.000
```

# Example

```
partSummary(out1.2, 8)

Defined Parameters:
                   Estimate  Std.Err  z-value  P(>|z|)
    ab1               0.016    0.014    1.093    0.274
    ab2               0.016    0.014    1.093    0.274
    ab3               0.012    0.029    0.412    0.680
    totalIE           0.043    0.042    1.038    0.299
```

# Example

```r
## Conduct a chi-squared difference test:
chiDiff <- fitMeasures(out1.2)["chisq"] - fitMeasures(out1.1)["chisq"]
dfDiff  <- fitMeasures(out1.2)["df"] - fitMeasures(out1.1)["df"]

pchisq(chiDiff, dfDiff, lower = FALSE)

      chisq
0.009440075
```

# Example

```
## Same test as above using bootstrapping:
mod1.3 <- "
policy ~ b1*sysRac + b2*indRac + b3*merit + cp*polAffil
sysRac ~ a1*polAffil
indRac ~ a2*polAffil
merit ~ a3*polAffil

sysRac ~~ indRac + merit
indRac ~~ merit

ab1 := a1*b1
ab2 := a2*b2
ab3 := a3*b3
totalIE := ab1 + ab2 + ab3

test1 := ab2 - ab1
"

out1.3 <- sem(mod1.3, data = dat1, se = "boot", bootstrap = 5000)
```

# Example

```
## Look at results:
partSummary(out1.3, 5)

Regressions:
                   Estimate  Std.Err  z-value  P(>|z|)
  policy ~
    sysRac    (b1)    0.601    0.141    4.253    0.000
    indRac    (b2)    0.143    0.108    1.324    0.186
    merit     (b3)   -0.036    0.150   -0.242    0.809
    polAffil  (cp)    0.125    0.077    1.629    0.103
  sysRac ~
    polAffil  (a1)    0.170    0.064    2.653    0.008
  indRac ~
    polAffil  (a2)   -0.004    0.079   -0.054    0.957
  merit ~
    polAffil  (a3)   -0.266    0.060   -4.465    0.000
```

# Example

```
partSummary(out1.3, 6:7)

Covariances:
                   Estimate  Std.Err  z-value  P(>|z|)
 .sysRac ~~
   .indRac           -0.076    0.100   -0.758    0.448
   .merit            -0.217    0.092   -2.361    0.018
 .indRac ~~
   .merit             0.154    0.098    1.580    0.114

Variances:
                   Estimate  Std.Err  z-value  P(>|z|)
   .policy            0.963    0.176    5.471    0.000
   .sysRac            0.755    0.107    7.050    0.000
   .indRac            1.188    0.152    7.833    0.000
   .merit             0.719    0.112    6.424    0.000
```

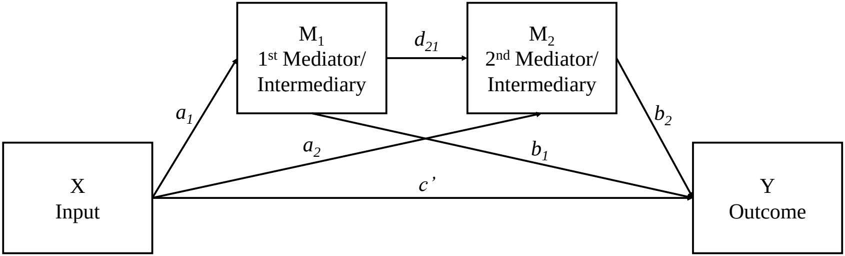
# Example

```
partSummary(out1.3, 8)

Defined Parameters:
                   Estimate  Std.Err  z-value  P(>|z|)
    ab1               0.102    0.044    2.305    0.021
    ab2              -0.001    0.015   -0.041    0.967
    ab3               0.010    0.041    0.235    0.814
    totalIE           0.111    0.052    2.134    0.033
    test1            -0.103    0.048   -2.131    0.033
```

# Example

```
parameterEstimates(out1.3, boot.ci.type = "bca.simple") %>%
    select(c("label", "est", "ci.lower", "ci.upper")) %>%
    tail(5)

     label    est ci.lower ci.upper
16     ab1  0.102    0.031    0.208
17     ab2 -0.001   -0.037    0.028
18     ab3  0.010   -0.076    0.087
19 totalIE  0.111    0.013    0.216
20   test1 -0.103   -0.214   -0.020
```

# Serial Multiple Mediation



# Serial Multiple Mediation

To get all of the information in the preceding diagram, we need to estimate three equations:

$$
\begin{aligned}
Y &= i_Y + b_1 M_1 + b_2 M_2 + c' X + e_Y \\
M_2 &= i_{M2} + d_{21} M_1 + a_2 X + e_{M2} \\
M_1 &= i_{M1} + a_1 X + e_{M1}
\end{aligned}
$$

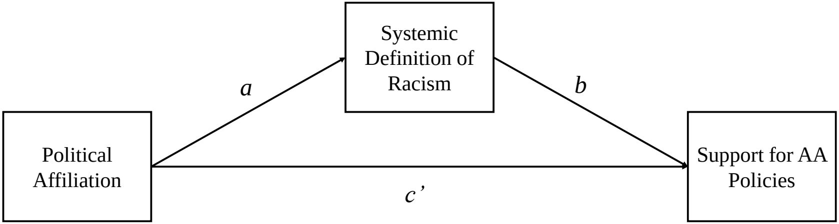
As with parallel mediator models, a serial mediator model with $K$ mediator variables will required $K + 1$ separate equations.

Again, path modeling can make this task much simpler.

- Also allows us to fit more parsimonious, restricted models.

# Serial Multiple Mediation

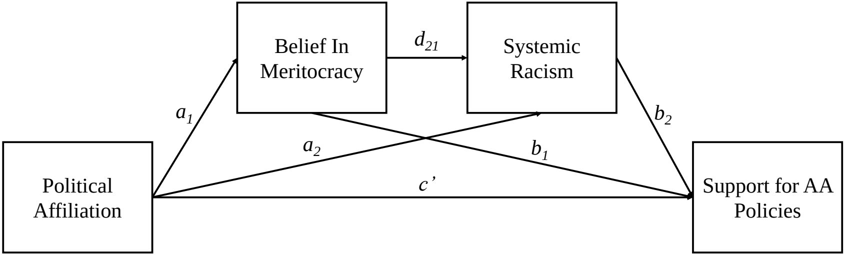
Okay, back to our simple mediation example:




QUESTION: What could be mediating the $a$ path?

# Serial Multiple Mediation

POTENTIAL ANSWER:



# A Quick Note on Inference

Parallel multiple mediation operates much like a number of combined simple mediation models.

- Serial multiple mediation is not so straight-forward.

In serial multiple mediation:

- Every possible path from $X$ to $Y$ that passes through, at least, one mediator is a specific indirect effect.
    - With the saturated two-mediator model shown above, we have: $IE_{spec} = \{a_1b_1, a_2b_2, a_1d_{21}b_2\}$
- The *Total Indirect Effect* is, again, equal to the sum of all the specific indirect effects: $IE_{tot} = \sum_{k=1}^{|\{IE_{spec}\}|} IE_{spec,k}$.
- The *Total Effect* is equal to the direct effect plus the total indirect effect: $c = c' + IE_{tot}$

# A Quick Note on Inference

Inference for the specific indirect effects is basically the same as it is for the sole indirect effect in simple mediation.

- CAVEAT: Normal-theory, Sobel-Type, standard errors for the specific indirect effects that involve more than two constituent paths can be very complex.
  - This isn't really a problem since you should always use bootstrapping, anyway!

# Example

```
## Serial Multiple Mediator Model:
mod2.1 <- "
policy ~ b1*merit + b2*sysRac + cp*polAffil
sysRac ~ d21*merit + a2*polAffil
merit ~ a1*polAffil

ab1 := a1*b1
ab2 := a2*b2
fullIE := a1*d21*b2
totalIE := ab1 + ab2 + fullIE
"

out2.1 <- sem(mod2.1, data = dat1, se = "boot", bootstrap = 5000)
```

# Example

```
## Check the results:
partSummary(out2.1, 5)

Regressions:
                   Estimate  Std.Err  z-value  P(>|z|)
  policy ~
    merit     (b1)   -0.008    0.147   -0.051    0.959
    sysRac    (b2)    0.595    0.145    4.111    0.000
    polAffil  (cp)    0.134    0.077    1.738    0.082
  sysRac ~
    merit    (d21)   -0.301    0.112   -2.695    0.007
    polAffil  (a2)    0.090    0.073    1.226    0.220
  merit ~
    polAffil  (a1)   -0.266    0.060   -4.409    0.000
```

# Example

```
partSummary(out2.1, 6:7)

Variances:
                   Estimate  Std.Err  z-value  P(>|z|)
   .policy            0.987    0.167    5.923    0.000
   .sysRac            0.689    0.092    7.482    0.000
   .merit             0.719    0.112    6.440    0.000

Defined Parameters:
                   Estimate  Std.Err  z-value  P(>|z|)
    ab1               0.002    0.040    0.049    0.961
    ab2               0.053    0.044    1.203    0.229
    fullIE            0.048    0.027    1.799    0.072
    totalIE           0.103    0.048    2.130    0.033
```

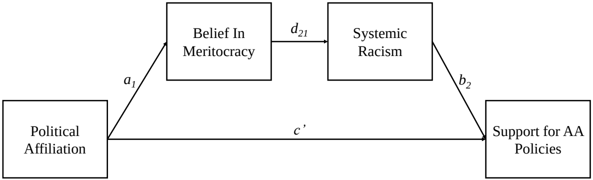
# Example

```
parameterEstimates(out2.1, boot.ci.type = "bca.simple") %>%
    select(c("label", "est", "ci.lower", "ci.upper")) %>%
    tail(4)

     label   est ci.lower ci.upper
11     ab1 0.002   -0.079    0.080
12     ab2 0.053   -0.029    0.150
13  fullIE 0.048    0.012    0.124
14 totalIE 0.103    0.014    0.206
```

# Restricted Models

In the preceding example, the $a_2$ and $b_1$ paths and the specific indirect effects $a_1 b_1$ and $a_2 b_2$ were all non-significant.

- There is a school of thinking that would prescribe constraining the $a_2$ and $b_1$ paths to zero as in:




- This model will ascribe a larger effect size to $a_1 d_{21} b_2$ since it must convey all of the indirect influence of $X$ on $Y$.

# Example

```r
mod2.2 <- "
policy ~ cp*polAffil + b2*sysRac
merit ~ a1*polAffil
sysRac ~ d21*merit

fullIE := a1*d21*b2
"

out2.2 <- sem(mod2.2, data = dat1, se = "boot", bootstrap = 5000)
```

# Example

```
partSummary(out2.2, 5:7)

Regressions:
                   Estimate  Std.Err  z-value  P(>|z|)
  policy ~
    polAffil  (cp)    0.135    0.083    1.635    0.102
    sysRac    (b2)    0.597    0.135    4.415    0.000
  merit ~
    polAffil  (a1)   -0.266    0.061   -4.385    0.000
  sysRac ~
    merit    (d21)   -0.367    0.095   -3.864    0.000

Variances:
                   Estimate  Std.Err  z-value  P(>|z|)
   .policy            0.987    0.166    5.958    0.000
   .merit             0.719    0.112    6.416    0.000
   .sysRac            0.705    0.093    7.578    0.000

Defined Parameters:
                   Estimate  Std.Err  z-value  P(>|z|)
    fullIE            0.058    0.025    2.336    0.019
```

# Example

```
parameterEstimates(out2.2, boot.ci.type = "bca.simple") %>%
    select(c("label", "est", "ci.lower", "ci.upper")) %>%
    filter(label != "")

   label    est ci.lower ci.upper
1     cp  0.135   -0.042    0.284
2     b2  0.597    0.317    0.849
3     a1 -0.266   -0.385   -0.146
4    d21 -0.367   -0.541   -0.170
5 fullIE  0.058    0.020    0.120
```

# Example

As in parallel multiple mediation, we can test for differences in the specific indirect effects of a serial multiple mediator model:

```
mod2.3 <- "
policy ~ cp*polAffil + b1*merit + b2*sysRac
merit ~ a1*polAffil
sysRac ~ a2*polAffil + d21*merit

ab1 := a1*b1
ab2 := a2*b2
fullIE := a1*d21*b2
totalIE := ab1 + ab2 + fullIE

fullIE == ab1
fullIE == ab2
"

out2.3 <- sem(mod2.3, data = dat1)
```

# Example

```
partSummary(out2.3, 5)

Regressions:
                   Estimate  Std.Err  z-value  P(>|z|)
  policy ~
    polAffil  (cp)    0.108    0.074    1.469    0.142
    merit     (b1)   -0.150    0.046   -3.243    0.001
    sysRac    (b2)    0.521    0.113    4.624    0.000
  merit ~
    polAffil  (a1)   -0.271    0.057   -4.769    0.000
  sysRac ~
    polAffil  (a2)    0.078    0.023    3.364    0.001
    merit    (d21)   -0.287    0.073   -3.925    0.000
```

# Example

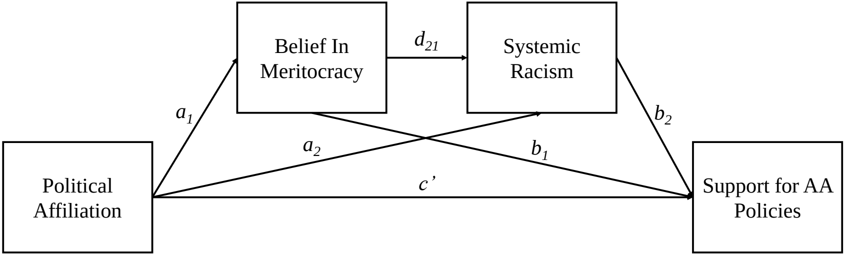
```
partSummary(out2.3, 6:7)

Variances:
                   Estimate  Std.Err  z-value  P(>|z|)
   .policy            1.001    0.152    6.595    0.000
   .merit             0.719    0.109    6.595    0.000
   .sysRac            0.690    0.105    6.595    0.000

Defined Parameters:
                   Estimate  Std.Err  z-value  P(>|z|)
    ab1               0.041    0.014    2.873    0.004
    ab2               0.041    0.014    2.873    0.004
    fullIE            0.041    0.014    2.873    0.004
    totalIE           0.122    0.042    2.873    0.004
```

# Example

```r
## Conduct a chi-squared difference test:
chiDiff <- fitMeasures(out2.3)["chisq"] - fitMeasures(out2.1)["chisq"]
dfDiff  <- fitMeasures(out2.3)["df"] - fitMeasures(out2.1)["df"]

pchisq(chiDiff, dfDiff, lower = FALSE)

    chisq
0.5131246
```

# Serial Multiple Mediation

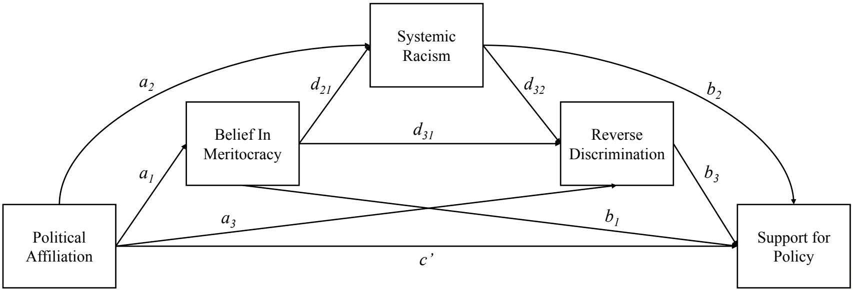
OK. We've supported an interesting hypothesis with the following model, but why stop there?




**QUESTION:** What might mediated the $b_2$ path?

# Serial Multiple Mediation

## POTENTIAL ANSWER:



# Serial Multiple Mediation

QUESTION: How many equations do we need to get the information in the preceding diagram?

# Serial Multiple Mediation

QUESTION: How many equations do we need to get the information in the preceding diagram?

$$
\begin{aligned}
Policy &= i_Y + b_1 Merit + b_2 SysRac + b_3 RevDisc + c' PolAff + e_Y \\
RevDisc &= i_{M3} + d_{31} Merit + d_{32} SysRac + a_3 PolAff + e_{M3} \\
SysRac &= i_{M2} + d_{21} Merit + a_2 PolAff + e_{M2} \\
Merit &= i_{M1} + a_1 PolAff + e_{M1}
\end{aligned}
$$

Which produces the following set of specific indirect effects:

- $a_1 b_1$
- $a_2 b_2$
- $a_3 b_3$
- $a_1 d_{31} b_3$
- $a_1 d_{21} b_2$
- $a_2 d_{32} b_3$
- $a_1 d_{21} d_{32} b_3$

# Example

```r
## Serial Multiple Mediator Model with 3 Mediators:
mod3.1 <- "
policy ~ b1*merit + b2*sysRac + b3*revDisc + cp*polAffil
revDisc ~ d31*merit + d32*sysRac + a3*polAffil
sysRac ~ d21*merit + a2*polAffil
merit ~ a1*polAffil

ab1 := a1*b1
ab2 := a2*b2
ab3 := a3*b3

partIE1 := a1*d31*b3
partIE2 := a1*d21*b2
partIE3 := a2*d32*b3

fullIE := a1*d21*d32*b3

totalIE := ab1 + ab2 + ab3 + partIE1 + partIE2 + partIE3 + fullIE
"

out3.1 <- sem(mod3.1, data = dat1, se = "boot", bootstrap = 5000)
```

# Example

```
partSummary(out3.1, 5)

Regressions:
                   Estimate  Std.Err  z-value  P(>|z|)
  policy ~
    merit     (b1)    0.005    0.146    0.034    0.973
    sysRac    (b2)    0.589    0.149    3.958    0.000
    revDisc   (b3)   -0.026    0.080   -0.328    0.743
    polAffil  (cp)    0.130    0.079    1.632    0.103
  revDisc ~
    merit    (d31)    0.473    0.192    2.471    0.013
    sysRac   (d32)   -0.196    0.242   -0.810    0.418
    polAffil  (a3)   -0.149    0.130   -1.146    0.252
  sysRac ~
    merit    (d21)   -0.301    0.111   -2.706    0.007
    polAffil  (a2)    0.090    0.071    1.261    0.207
  merit ~
    polAffil  (a1)   -0.266    0.061   -4.399    0.000
```

# Example

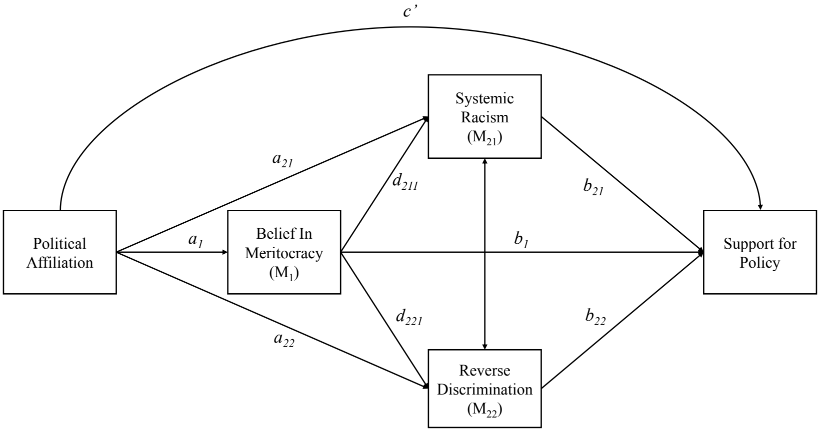
```
partSummary(out3.1, 6:7)

Variances:
                   Estimate  Std.Err  z-value  P(>|z|)
   .policy            0.985    0.166    5.932    0.000
   .revDisc           2.361    0.307    7.682    0.000
   .sysRac            0.689    0.090    7.630    0.000
   .merit             0.719    0.112    6.406    0.000

Defined Parameters:
                   Estimate  Std.Err  z-value  P(>|z|)
    ab1              -0.001    0.040   -0.033    0.974
    ab2               0.053    0.044    1.207    0.228
    ab3               0.004    0.016    0.242    0.808
    partIE1           0.003    0.013    0.265    0.791
    partIE2           0.047    0.027    1.782    0.075
    partIE3           0.000    0.003    0.151    0.880
    fullIE            0.000    0.002    0.184    0.854
    totalIE           0.107    0.052    2.043    0.041
```

# Example

```
parameterEstimates(out3.1, boot.ci.type = "bca.simple") %>%
    select(c("label", "est", "ci.lower", "ci.upper")) %>%
    filter(label != "")

      label    est ci.lower ci.upper
1        b1  0.005   -0.293    0.284
2        b2  0.589    0.282    0.873
3        b3 -0.026   -0.186    0.129
4        cp  0.130   -0.034    0.278
5       d31  0.473    0.086    0.845
6       d32 -0.196   -0.639    0.287
7        a3 -0.149   -0.413    0.101
8       d21 -0.301   -0.517   -0.085
9        a2  0.090   -0.063    0.222
10       a1 -0.266   -0.391   -0.153
11      ab1 -0.001   -0.082    0.078
12      ab2  0.053   -0.028    0.151
13      ab3  0.004   -0.015    0.059
14 partIE1  0.003   -0.015    0.037
15 partIE2  0.047    0.011    0.119
16 partIE3  0.000   -0.002    0.018
17  fullIE  0.000   -0.002    0.010
18 totalIE [illegible]
```

# Hybrid Multiple Mediation

We can also combine parallel and serial mediation models:



# Example

```
## Hybrid Multiple Mediator Model:
mod4.1 <- "
policy ~ b1*merit + b21*sysRac + b22*revDisc + cp*polAffil
sysRac ~ d211*merit + a21*polAffil
revDisc ~ d221*merit + a22*polAffil
merit ~ a1*polAffil

sysRac ~~ revDisc

ab1 := a1*b1
ab21 := a21*b21
ab22 := a22*b22

fullIE21 := a1*d211*b21
fullIE22 := a1*d221*b22

totalIE := ab1 + ab21 + ab22 + fullIE21 + fullIE22
"

out4.1 <- sem(mod4.1, data = dat1, se = "boot", bootstrap = 500)
```

# Example

```
partSummary(out4.1, 5)

Regressions:
                   Estimate  Std.Err  z-value  P(>|z|)
  policy ~
    merit      (b1)    0.005    0.143    0.035    0.972
    sysRac    (b21)    0.589    0.149    3.969    0.000
    revDisc   (b22)   -0.026    0.081   -0.327    0.743
    polAffl    (cp)    0.130    0.082    1.584    0.113
  sysRac ~
    merit    (d211)   -0.301    0.111   -2.720    0.007
    polAffl   (a21)    0.090    0.070    1.277    0.201
  revDisc ~
    merit    (d221)    0.532    0.182    2.922    0.003
    polAffl   (a22)   -0.167    0.127   -1.311    0.190
  merit ~
    polAffl    (a1)   -0.266    0.062   -4.311    0.000
```

# Example

```
partSummary(out4.1, 6:7)

Covariances:
                   Estimate  Std.Err  z-value  P(>|z|)
 .sysRac ~~
   .revDisc          -0.135    0.157   -0.859    0.390

Variances:
                   Estimate  Std.Err  z-value  P(>|z|)
   .policy            0.985    0.167    5.917    0.000
   .sysRac            0.689    0.087    7.883    0.000
   .revDisc           2.388    0.289    8.249    0.000
   .merit             0.719    0.116    6.200    0.000
```

# Example

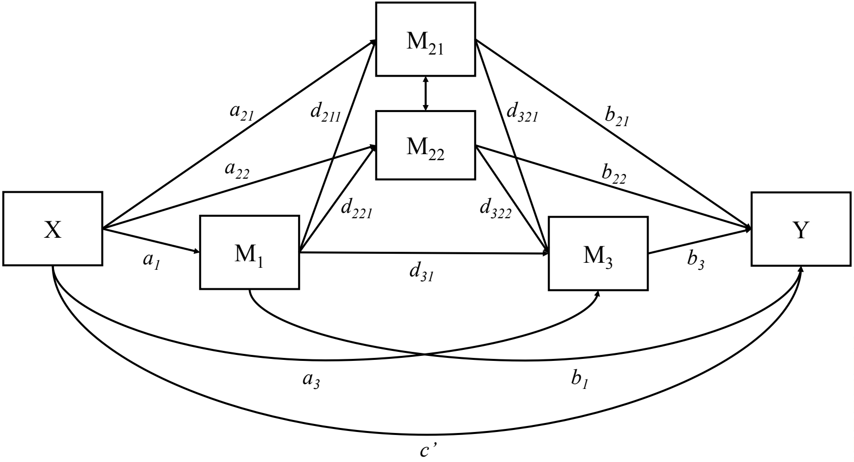
```
partSummary(out4.1, 8)

Defined Parameters:
                   Estimate  Std.Err  z-value  P(>|z|)
    ab1              -0.001    0.038   -0.035    0.972
    ab21              0.053    0.045    1.177    0.239
    ab22              0.004    0.018    0.249    0.803
    fullIE21          0.047    0.027    1.751    0.080
    fullIE22          0.004    0.012    0.307    0.759
    totalIE           0.107    0.052    2.061    0.039
```

# Example

```
parameterEstimates(out4.1, boot.ci.type = "bca.simple") %>%
    select(c("label", "est", "ci.lower", "ci.upper")) %>%
    filter(label != "")

      label    est ci.lower ci.upper
1        b1  0.005   -0.285    0.294
2       b21  0.589    0.296    0.862
3       b22 -0.026   -0.181    0.133
4        cp  0.130   -0.034    0.302
5      d211 -0.301   -0.516   -0.059
6       a21  0.090   -0.064    0.214
7      d221  0.532    0.169    0.880
8       a22 -0.167   -0.390    0.087
9        a1 -0.266   -0.396   -0.153
10      ab1 -0.001   -0.078    0.074
11     ab21  0.053   -0.036    0.144
12     ab22  0.004   -0.017    0.058
13 fullIE21  0.047    0.012    0.133
14 fullIE22  0.004   -0.016    0.036
15  totalIE  0.107    0.007    0.209
```

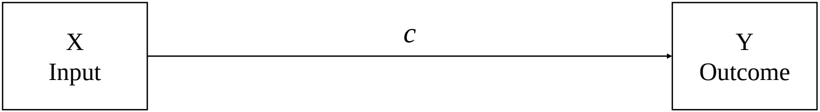
# Practice

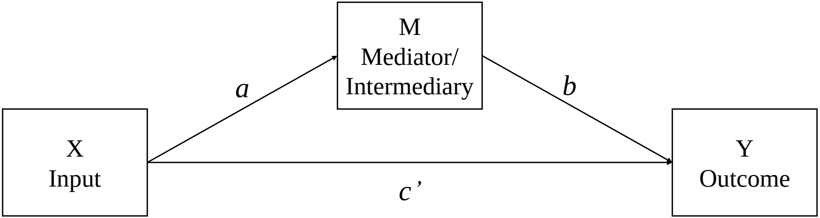
List all of the specific indirect effects present in this model:



# Boring Model

So far, all of our models have looked something like:

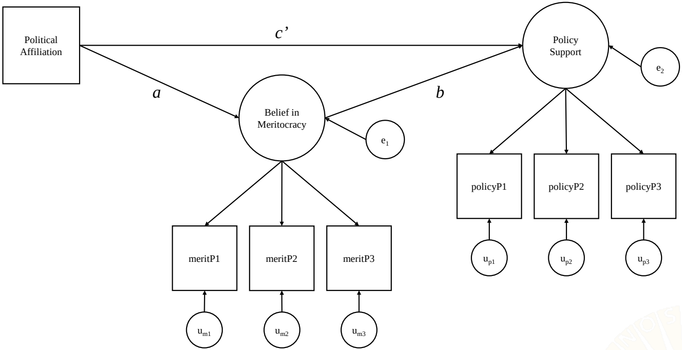



But there is no reason that we need to restrict ourselves to mucking around with

# Better Model

We can (and should) test for indirect effects using full SEMs such as:




Measurement error can be a big problem for mediation analysis, so latent variable modeling is highly recommended.

# Example

```
dat1 <- readRDS("../data/adamsKlpsData.rds") %>% select(-merit, -policy)

## Specify the CFA model:
mod5.1 <- "
merit =~ meritP1 + meritP2 + meritP3
policy =~ policyP1 + policyP2 + policyP3
"

## Fit the CFA and check model:
out5.1 <- cfa(mod5.1, data = dat1, std.lv = TRUE)

## Check model fit:
fitMeasures(out5.1,
            c("chisq", "df", "pvalue", "cfi", "tli", "rmsea", "srmr")
            )

 chisq      df  pvalue    cfi    tli  rmsea   srmr
16.869   8.000   0.031  0.922  0.853  0.113  0.065
```

# Example

```
partSummary(out5.1, 5)

Latent Variables:
                   Estimate  Std.Err  z-value  P(>|z|)
  merit =~
    meritP1           0.690    0.134    5.155    0.000
    meritP2           0.968    0.142    6.830    0.000
    meritP3           0.748    0.137    5.458    0.000
  policy =~
    policyP1          0.851    0.186    4.570    0.000
    policyP2          0.996    0.167    5.967    0.000
    policyP3          1.121    0.177    6.339    0.000
```

# Example

```
partSummary(out5.1, 6:7)

Covariances:
                   Estimate  Std.Err  z-value  P(>|z|)
  merit ~~
    policy           -0.336    0.131   -2.563    0.010

Variances:
                   Estimate  Std.Err  z-value  P(>|z|)
   .meritP1           0.865    0.165    5.248    0.000
   .meritP2           0.445    0.201    2.211    0.027
   .meritP3           0.833    0.172    4.857    0.000
   .policyP1          1.836    0.324    5.671    0.000
   .policyP2          0.942    0.256    3.683    0.000
   .policyP3          0.857    0.297    2.882    0.004
    merit             1.000
    policy            1.000
```

# Example

```r
## Specify the structural model:
mod5.2 <- "
merit =~ meritP1 + meritP2 + meritP3
policy =~ policyP1 + policyP2 + policyP3

policy ~ b*merit + polAffil
merit ~ a*polAffil

ab := a*b
"

## Fit the structural model and test the indirect effect:
out5.2 <-
    sem(mod5.2, data = dat1, std.lv = TRUE, se = "boot", bootstrap = 2000)
```

# Example

```
partSummary(out5.2, 5:6)

Latent Variables:
                   Estimate  Std.Err  z-value  P(>|z|)
  merit =~
    meritP1           0.545    0.501    1.088    0.277
    meritP2           0.858    0.129    6.658    0.000
    meritP3           0.609    0.119    5.117    0.000
  policy =~
    policyP1          0.799    0.187    4.266    0.000
    policyP2          0.924    1.778    0.520    0.603
    policyP3          1.001    1.342    0.746    0.456

Regressions:
                   Estimate  Std.Err  z-value  P(>|z|)
  policy ~
    merit      (b)   -0.195    0.204   -0.955    0.340
    polAffil          0.169    0.134    1.262    0.207
  merit ~
    polAffil   (a)   -0.411    0.098   -4.177    0.000
```

# Example

```
partSummary(out5.2, 7:8)

Variances:
                   Estimate  Std.Err  z-value  P(>|z|)
   .meritP1           0.922   11.008    0.084    0.933
   .meritP2           0.341    0.223    1.529    0.126
   .meritP3           0.869    0.179    4.848    0.000
   .policyP1          1.801    0.336    5.365    0.000
   .policyP2          0.918   88.978    0.010    0.992
   .policyP3          0.922   53.003    0.017    0.986
   .merit             1.000
   .policy            1.000

Defined Parameters:
                   Estimate  Std.Err  z-value  P(>|z|)
    ab                0.080    0.089    0.901    0.368
```

# Example

```
parameterEstimates(out5.2, boot.ci.type = "bca.simple") %>%
    select(c("label", "est", "ci.lower", "ci.upper")) %>%
    filter(label != "")

  label    est ci.lower ci.upper
1     b -0.195   -0.615    0.190
2     a -0.411   -0.634   -0.238
3    ab  0.080   -0.072    0.273
```

# Interpretation of Indirect Effects

Indirect effects are composed parameters, but they can be interpreted independently of their constituent paths.

- The $X \rightarrow M \rightarrow Y$ indirect effect, $ab$, is interpreted as:
  - The expected change in $Y$ for a unit change in $X$ that is transmitted indirectly through $M$.
  - For a unit change in $X$, $Y$ is expected to change by $ab$ units, indirectly through $M$.
  - Participants who differ by one unit on $X$ are expect to differ by $ab$ units on $Y$ as a results of the effect of $X$ on $M$ which, in turn, affects $Y$.

- The interpretation/scaling of the indirect effect is entirely defined by the input, $X$, and outcome, $Y$.
  - The scaling of the intermediary variable, $M$, does not affect the interpretation of the indirect effect.

# References

Baron, R. M., & Kenny, D. A. (1986). The moderator–mediator variable distinction in social psychological research: Conceptual, strategic, and statistical considerations. *Journal of Personality and Social Psychology*, *51*(6), 1173.

Efron, B. (1979). Bootstrap methods: Another look at the jackknife. *The Annals of Statistics*, *7*(1), 1–26. doi: 10.1214/aos/1176344552

Sobel, M. E. (1982). Asymptotic confidence intervals for indirect effects in structural equation models. *Sociological Methodology*, *13*(1982), 290–312.