SEM with lavaan: syntax overview

Ben Kite[1], Paul Johnson[1], and others[2]

[1]Center for Research Methods and Data Analysis [2]Previous GRAs, including but not limited to Terry Jorgensen, Sunthud Pornprasertmanit, Jared Harpole

2017

Outline

There is no SEM in R base

- Neither R nor any of the required or recommended packages distributed with it include structural equation models. But, . . .
- R does include mathematical functions with which SEM estimators can be created

Competing packages

- `sem()` in the "car" package by John Fox. Nearly as old as R itself
- `lavaan` , a package prepared by Yves Rosseel, who intends a function-by-function replication of the results in Mplus
- `OpenMX` , a long-standing matrix calculation framework spearheaded by Steven Bolker, which has been re-written as an R package.
- Various other R packages exist, either as wholesale replacements (`lava`) or as supplementary tools (`semPlot`).

Here we are focused on lavaan

- lavaan is the closest to a "full suite" of SEM tools needed by researchers
- lavaan includes many different estimation algorithms (FIML, WLSMV, etc)
- There is a comprehensive essay,
  Rosseel, Yves (2012). lavaan: An R Package for Structural Equation Modeling. *Journal of Statistical Software*, 48(2), 1 - 36. `doi:http://dx.doi.org/10.18637/jss.v048.i02`
- Its author, Yves Rosseel, has demonstrated an extremely high willingness to interact with users and design new features when needs arise

Use affect.rds from summeR-3-2-lm

```
dat <- readRDS("../data/affect.rds")
```

```
head(dat)
```

```
  Agency1 Agency2 Agency3 Intrin1 Intrin2 Intrin3 Extrin1
1  3.5000  4.0000  4.0000  4.0000     4.0       4  1.0000
2  2.5000  3.1667  3.0000  3.2123     2.0       3  1.8333
3  1.8333  2.0000  1.5000  3.0000     3.0       2  1.0000
4  2.7714  3.0602  2.3639  3.1337     4.0       3  1.0774
5  3.1667  3.3333  2.8333  3.5000     4.0       4  1.8333
6  2.3333  2.8333  2.3333  3.0000     2.5       3  3.0588
  Extrin2 Extrin3 PosAFF1 PosAFF2 PosAFF3 NegAFF1 NegAFF2
1  1.0000  1.5000  4.0000     4.0     4.0     1.0  1.0000
2  2.6667  1.8333  3.0000     3.5     2.5     1.5  1.6858
3  1.0000  1.0000  3.0184     2.5     3.0     1.0  1.0000
4  1.1667  1.0000  3.0000     2.5     3.0     2.5  2.5000
5  2.0000  1.8333  3.7804     3.5     3.0     2.5  2.0000
6  2.4125  2.6667  4.0000     3.0     3.0     2.0  1.5000
  NegAFF3 Sex gender ethnicity     race
1     1.0   1   male  Hispanic Nonwhite
2     1.5   1   male     White    White
3     1.0   1   male     White    White
4     1.5   1   male     White    White
5     3.0   1   male     White    White
6     2.0   1   male     White    White
```

Quick Jargon review

Indicators: The observed (aka “manifest”) variables

Latent variables: (aka “factors”, “latent constructs” or “common factors”): unobserved variables thought to be the things we are truly interested in. We’d really like to study the relationship among them, but we are unable to do so.

factor loadings: the coefficients which indicate how tightly an indicator is linked to the latent variable.

lavaan's "model fitting" functions

- The key modeling functions are
  - `lavaan()` : a general model fitter
  - `sem()` : a convenience for fitting structural equation models, receives the input and reformats it to send to lavaan()
  - `cfa()` : A convenience that translates input and uses `lavaan()` function in the background

Seemingly overwhelming number of arguments

- The help pages for `sem` and `cfa` list a lot of arguments

```
library(lavaan)
methods::formalArgs(sem)
```

```
 [1] "model"                 "data"
 [3] "meanstructure"         "conditional.x"
 [5] "fixed.x"               "orthogonal"
 [7] "std.lv"                "parameterization"
 [9] "std.ov"                "missing"
[11] "ordered"               "sample.cov"
[13] "sample.cov.rescale"    "sample.mean"
[15] "sample.nobs"           "ridge"
[17] "group"                 "group.label"
[19] "group.equal"           "group.partial"
[21] "group.w.free"          "cluster"
[23] "constraints"           "estimator"
[25] "likelihood"            "link"
[27] "information"           "se"
[29] "test"                  "bootstrap"
[31] "mimic"                 "representation"
[33] "do.fit"                "control"
```

Seemingly overwhelming number of arguments ...

```
[35] "WLS.V"               "NACOV"
[37] "zero.add"            "zero.keep.margins"
[39] "zero.cell.warn"      "start"
[41] "check"               "verbose"
[43] "warn"                "debug"
```

- Our mission here is to sort though these, emphasize the important ones, understand that others exist when necessary

The most important arguments

- `model` : a formula (character variable representing a formula)
- `data` : name of data.frame containing variables
- `estimator` : model estimator, defaults to "ML"
- `ordered` : vector of variables to be treated as ordered-categorical

Outline

Start with R regression syntax

- A regression in R:

```
mod1 <- lm(PosAFF1 ~ Agency1 + Intrin1, data =
    dat)
summary(mod1)
```

```
Call:
lm(formula = PosAFF1 ~ Agency1 + Intrin1, data = dat)

Residuals:
     Min       1Q   Median       3Q      Max
-2.17693 -0.35887  0.05093  0.55123  1.12382

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  2.05775    0.18125  11.353  < 2e-16 ***
Agency1      0.18594    0.06581   2.825  0.00497 **
Intrin1      0.20778    0.04414   4.707 3.54e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.6363 on 377 degrees of freedom
```

Start with R regression syntax ...

```
Multiple R^2:  0.09673,  Adjusted R^2:  0.09194
F-statistic: 20.19 on 2 and 377 DF,  p-value: 4.693e-09
```

The formula "PosAFF1 ~ Agency1 + Intrin1" is *in* the function call.

- A larger, more elaborate formula can be written as a separate piece, `fmla2`

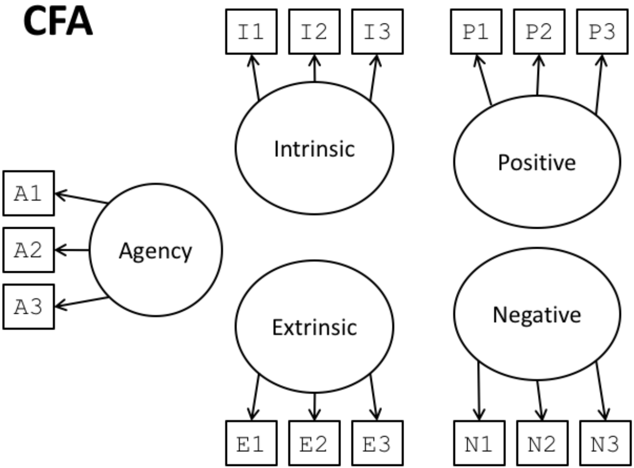
```
fmla2 <- "PosAFF1 ~ Agency1 + Intrin1"
```

- `fmla2` is a "character string" variable, but the `lm` function understands what to do

```
mod2 <- lm(fmla2, data = dat)
```

- Because structural equation models are almost always "larger" formulas, they are almost always written out as character structures in that style.

Make a lavaan formula for this



The mathematical formula

- The mathematical model: each indicator is predicted from a latent variable.

$$
\begin{aligned}
Agency1_i &= \alpha_{11} + \lambda_{11} Agency_i + \varepsilon_{11i} \\
Agency2_i &= \alpha_{12} + \lambda_{12} Agency_i + \varepsilon_{12i} \\
Agency3_i &= \alpha_{13} + \lambda_{13} Agency_i + \varepsilon_{13i}
\end{aligned}
$$

$$
\begin{aligned}
Extrin1_i &= \alpha_{21} + \lambda_{21} Extrinsic_i + \varepsilon_{21i} \\
Extrin2_i &= \alpha_{22} + \lambda_{22} Extrinsic_i + \varepsilon_{22i} \\
Extrin3_i &= \alpha_{23} + \lambda_{23} Extrinsic_i + \varepsilon_{23i}
\end{aligned}
$$

Note the book-keeping

- α (alpha) is the "intercept" (expected value when LV is 0)
- λ (beta) is the factor loading
- first subscript is the number of the latent variable, 1 = Agency, 2 = Extrinsic
- second subscript is the number of the indicator

lavaan introduces symbol "=~"

- In lavaan formulas, the syntax for linkage between measured and unmeasured variables is like so:

```
a_latent_variable =~ indicator1 + indicator2 +
    indicator3
```

- We name "a_latent_variable" whatever we want
- "indicator1", "indicator2", "indicator3" must be a variables named in the data frame

```
Agency =~ Agency1 + Agency2 + Agency3
```

- Note how, while the mathematical formula has $Agency1_i = \alpha + \lambda Agency_i + \varepsilon$, the lavaan formula syntax has the sides reversed

Another lavaan symbol "∼∼"

- To estimate the covariance between two latent variables, the symbol ∼∼ (that's two tilde) is used.
- For example, we allow the covariance between to latent variables by including this in our formula.

```
latent1 ~~ latent2
```

Example model syntax for Confirmatory Factor Analysis

```
cfa.mod <- '
 Agency =~ Agency1 + Agency2 + Agency3
 Intrinsic =~ Intrin1 + Intrin2 + Intrin3
 Extrinsic =~ Extrin1 + Extrin2 + Extrin3
 Positive =~ PosAFF1 + PosAFF2 + PosAFF3
 Negative =~ NegAFF1 + NegAFF2 + NegAFF3
 '
```

- You can use single or double quotes
- Line breaks are visual enhancement, which lavaan understands correctly as separate pieces
- Note, we do not include any `~~` symbols here (will return to that topic)

Could instead write out one element per indicator

```
cfa.mod <- '
 Agency =~ Agency1
 Agency =~ Agency2
 Agency =~ Agency3
 Intrinsic =~ Intrin1
 Intrinsic =~ Intrin2
 Intrinsic =~ Intrin3
 Extrinsic =~ Extrin1
 Extrinsic =~ Extrin2
 Extrinsic =~ Extrin3
 Positive =~ PosAFF1
 Positive =~ PosAFF2
 Positive =~ PosAFF3
 Negative =~ NegAFF1
 Negative =~ NegAFF2
 Negative =~ NegAFF3
```

Could instead write out one element per indicator ...

```
'
```

- That is verbose, we don't generally write it that way
- This may be easier in very large projects where functions are writing out the formula for us.

Estimate with cfa

```
cfa1 <- cfa(cfa.mod, data = dat)
```

- If there is no output, the model was estimated without errors
- As in other R model fitters, the `summary()` method is used to overview the results

```
summary(cfa1)
```

```
lavaan (0.5-22) converged normally after  66 iterations

  Number of observations                           380

  Estimator                                         ML
  Minimum Function Test Statistic              106.847
  Degrees of freedom                                80
  P-value (Chi-square)                           0.024

Parameter Estimates:

  Information                                 Expected
```

Estimate with cfa ...

```
  Standard Errors                              Standard

Latent Variables:
                    Estimate   Std.Err   z-value   P(>|z|)
  Agency =~
    Agency1            1.000
    Agency2            1.054     0.036    29.454     0.000
    Agency3            1.065     0.038    27.987     0.000
  Intrinsic =~
    Intrin1            1.000
    Intrin2            1.075     0.097    11.043     0.000
    Intrin3            1.138     0.096    11.832     0.000
  Extrinsic =~
    Extrin1            1.000
    Extrin2            1.177     0.077    15.356     0.000
    Extrin3            1.213     0.077    15.720     0.000
  Positive =~
    PosAFF1            1.000
    PosAFF2            1.060     0.049    21.607     0.000
    PosAFF3            1.110     0.051    21.963     0.000
  Negative =~
    NegAFF1            1.000
    NegAFF2            0.923     0.038    24.210     0.000
    NegAFF3            0.944     0.037    25.639     0.000
```

Estimate with cfa ...

```
Covariances:
                   Estimate  Std.Err  z-value  P(>|z|)
  Agency ~~
    Intrinsic         0.128    0.018    7.050    0.000
    Extrinsic         0.049    0.011    4.486    0.000
    Positive          0.080    0.016    5.152    0.000
    Negative          0.005    0.016    0.312    0.755
  Intrinsic ~~
    Extrinsic        -0.006    0.013   -0.440    0.660
    Positive          0.127    0.021    5.957    0.000
    Negative          0.007    0.021    0.341    0.733
  Extrinsic ~~
    Positive         -0.006    0.013   -0.502    0.616
    Negative          0.051    0.015    3.514    0.000
  Positive ~~
    Negative         -0.026    0.020   -1.264    0.206

Variances:
                   Estimate  Std.Err  z-value  P(>|z|)
   .Agency1           0.048    0.005    9.649    0.000
   .Agency2           0.036    0.005    7.604    0.000
   .Agency3           0.051    0.005    9.279    0.000
   .Intrin1           0.298    0.029   10.204    0.000
   .Intrin2           0.386    0.036   10.620    0.000
   .Intrin3           0.214    0.029    7.474    0.000
```

Estimate with cfa ...

```
   .Extrin1          0.080    0.009     8.924    0.000
   .Extrin2          0.123    0.013     9.465    0.000
   .Extrin3          0.103    0.012     8.264    0.000
   .PosAFF1          0.121    0.012     9.921    0.000
   .PosAFF2          0.104    0.012     8.615    0.000
   .PosAFF3          0.099    0.013     7.870    0.000
   .NegAFF1          0.107    0.012     9.202    0.000
   .NegAFF2          0.096    0.010     9.498    0.000
   .NegAFF3          0.067    0.009     7.369    0.000
    Agency           0.218    0.019    11.280    0.000
    Intrinsic        0.292    0.042     6.980    0.000
    Extrinsic        0.150    0.017     8.803    0.000
    Positive         0.324    0.032    10.066    0.000
    Negative         0.402    0.037    10.835    0.000
```

Why did we use cfa, not lavaan, to fit that?

- As explained in " `?cfa` ", the cfa function has some default settings that are customary for confirmatory factor analysis.
- The settings are:
  - Estimate observed variable intercepts, set the means of the latent variables to 0, fix the loading for the first indicator to 1 (unless std.lv = TRUE), and estimate the covariances between latent variables.
- A very important standard setting for `cfa` is that the covariances are estimated among the latent variables, even though we did not explicitly ask for them.

Variances among the latent variables: "∼∼"

- Reminder: covariance estimate is requested by `latent1` ∼∼ `latent2` .
- Did you notice in the output that the `cfa()` function estimated those for us, even though we did not ask for them?
- We could have asked, explicitly, for covariances:

```
cfa.mod2 <- '
 Agency =~ Agency1 + Agency2 + Agency3
 Intrinsic =~ Intrin1 + Intrin2 + Intrin3
 Extrinsic =~ Extrin1 + Extrin2 + Extrin3
 Positive =~ PosAFF1 + PosAFF2 + PosAFF3
 Negative =~ NegAFF1 + NegAFF2 + NegAFF3
 Agency ~~ Intrinsic
 Agency ~~ Extrinsic
 Agency ~~ Positive
```

Variances among the latent variables: "∼∼" ...

```
Agency ~~ Negative
Intrinsic ~~ Extrinsic
Intrinsic ~~ Positive
Intrinsic ~~ Negative
Extrinsic ~~ Positive
Extrinsic ~~ Negative
Positive ~~ Negative
'
```

```
cfa2 <- cfa(cfa.mod2, data = dat)
summary(cfa2)
```

Variances among the latent variables: "∼∼" ...

```
lavaan (0.5-22) converged normally after  66 iterations

  Number of observations                           380

  Estimator                                         ML
  Minimum Function Test Statistic              106.847
  Degrees of freedom                                80
  P-value (Chi-square)                           0.024

Parameter Estimates:

  Information                                 Expected
  Standard Errors                             Standard

Latent Variables:
                   Estimate  Std.Err  z-value  P(>|z|)
  Agency =~
    Agency1           1.000
    Agency2           1.054    0.036   29.454    0.000
    Agency3           1.065    0.038   27.987    0.000
  Intrinsic =~
    Intrin1           1.000
    Intrin2           1.075    0.097   11.043    0.000
    Intrin3           1.138    0.096   11.832    0.000
```

Variances among the latent variables: "~~" ...

```
  Extrinsic =~
    Extrin1          1.000
    Extrin2          1.177    0.077   15.356    0.000
    Extrin3          1.213    0.077   15.720    0.000
  Positive =~
    PosAFF1          1.000
    PosAFF2          1.060    0.049   21.607    0.000
    PosAFF3          1.110    0.051   21.963    0.000
  Negative =~
    NegAFF1          1.000
    NegAFF2          0.923    0.038   24.210    0.000
    NegAFF3          0.944    0.037   25.639    0.000

Covariances:
                  Estimate  Std.Err  z-value  P(>|z|)
  Agency ~~
    Intrinsic        0.128    0.018    7.050    0.000
    Extrinsic        0.049    0.011    4.486    0.000
    Positive         0.080    0.016    5.152    0.000
    Negative         0.005    0.016    0.312    0.755
  Intrinsic ~~
    Extrinsic       -0.006    0.013   -0.440    0.660
    Positive         0.127    0.021    5.957    0.000
    Negative         0.007    0.021    0.341    0.733
  Extrinsic ~~
```

Variances among the latent variables: "$\sim\sim$" ...

```
    Positive       -0.006    0.013   -0.502    0.616
    Negative        0.051    0.015    3.514    0.000
  Positive ~~
    Negative       -0.026    0.020   -1.264    0.206

Variances:
                 Estimate  Std.Err  z-value  P(>|z|)
   .Agency1         0.048    0.005    9.649    0.000
   .Agency2         0.036    0.005    7.604    0.000
   .Agency3         0.051    0.005    9.279    0.000
   .Intrin1         0.298    0.029   10.204    0.000
   .Intrin2         0.386    0.036   10.620    0.000
   .Intrin3         0.214    0.029    7.474    0.000
   .Extrin1         0.080    0.009    8.924    0.000
   .Extrin2         0.123    0.013    9.465    0.000
   .Extrin3         0.103    0.012    8.264    0.000
   .PosAFF1         0.121    0.012    9.921    0.000
   .PosAFF2         0.104    0.012    8.615    0.000
   .PosAFF3         0.099    0.013    7.870    0.000
   .NegAFF1         0.107    0.012    9.202    0.000
   .NegAFF2         0.096    0.010    9.498    0.000
   .NegAFF3         0.067    0.009    7.369    0.000
    Agency          0.218    0.019   11.280    0.000
    Intrinsic       0.292    0.042    6.980    0.000
    Extrinsic       0.150    0.017    8.803    0.000
```

Variances among the latent variables: "$\sim\sim$" ...

```
  Positive          0.324     0.032    10.066     0.000
  Negative          0.402     0.037    10.835     0.000
```

Outline

In lavaan, $\sim$ represents a regression

The symbol latent1 $\sim$ latent2 indicates that one latent variable is predicting another.

We are, in essence, asking SEM to calculate coefficients for a regression model:

$$latentA_i = \beta_0 + \beta_1 latentB_i + v_i$$

These slope coefficients represent “directionality”.

Rather than covarying, latentB predicts latentA (could be a causal relationship).

Coefficients can be named for future reference

In lavaan formulae, coefficients can be named

$$latentA_i \sim c(beta1) * latentB_i$$

Outline

Did you notice loadings = 1.0 in previous output?

- Scaling of latent variables and loading coefficients is our choice
- Previous method is known as the "marker variable" method (Lindell & Whitney, 2001), because for each latent variable a single variable was selected to determine the scale.
- Another commonly used method is to fix the variance of the latent variables at 1.0 (thus allowing the loadings to float freely)

Obtain "standardized latent variable" estimates

```
cfa3 <- cfa(cfa.mod, data = dat, std.lv = TRUE)
```

```
summary(cfa3)
```

```
lavaan (0.5-22) converged normally after  59 iterations

  Number of observations                           380

  Estimator                                         ML
  Minimum Function Test Statistic              106.847
  Degrees of freedom                                80
  P-value (Chi-square)                           0.024

Parameter Estimates:

  Information                                 Expected
  Standard Errors                             Standard

Latent Variables:
                   Estimate  Std.Err  z-value  P(>|z|)
  Agency =~
    Agency1           0.466    0.021   22.560    0.000
```

Obtain "standardized latent variable" estimates ...

```
    Agency2          0.492    0.021   23.774    0.000
    Agency3          0.497    0.022   22.815    0.000
  Intrinsic =~
    Intrin1          0.541    0.039   13.960    0.000
    Intrin2          0.581    0.043   13.489    0.000
    Intrin3          0.615    0.038   16.201    0.000
  Extrinsic =~
    Extrin1          0.388    0.022   17.606    0.000
    Extrin2          0.456    0.027   17.168    0.000
    Extrin3          0.471    0.026   18.100    0.000
  Positive =~
    PosAFF1          0.569    0.028   20.132    0.000
    PosAFF2          0.603    0.029   21.150    0.000
    PosAFF3          0.632    0.029   21.648    0.000
  Negative =~
    NegAFF1          0.634    0.029   21.670    0.000
    NegAFF2          0.585    0.027   21.457    0.000
    NegAFF3          0.598    0.026   22.805    0.000

Covariances:
                  Estimate  Std.Err  z-value  P(>|z|)
  Agency ~~
    Intrinsic        0.507    0.047   10.782    0.000
    Extrinsic        0.270    0.054    5.028    0.000
    Positive         0.302    0.051    5.950    0.000
```

Obtain "standardized latent variable" estimates ...

```
    Negative          0.017    0.055    0.312    0.755
  Intrinsic ~~
    Extrinsic        -0.028    0.063   -0.441    0.659
    Positive          0.414    0.052    7.918    0.000
    Negative          0.021    0.060    0.341    0.733
  Extrinsic ~~
    Positive         -0.029    0.058   -0.503    0.615
    Negative          0.209    0.056    3.764    0.000
  Positive ~~
    Negative         -0.071    0.056   -1.274    0.203

Variances:
                   Estimate  Std.Err  z-value  P(>|z|)
   .Agency1           0.048    0.005    9.649    0.000
   .Agency2           0.036    0.005    7.604    0.000
   .Agency3           0.051    0.005    9.278    0.000
   .Intrin1           0.298    0.029   10.204    0.000
   .Intrin2           0.386    0.036   10.620    0.000
   .Intrin3           0.214    0.029    7.474    0.000
   .Extrin1           0.080    0.009    8.924    0.000
   .Extrin2           0.123    0.013    9.465    0.000
   .Extrin3           0.103    0.012    8.264    0.000
   .PosAFF1           0.121    0.012    9.921    0.000
   .PosAFF2           0.104    0.012    8.615    0.000
   .PosAFF3           0.099    0.013    7.870    0.000
```

Obtain "standardized latent variable" estimates ...

```
   .NegAFF1          0.107    0.012    9.202    0.000
   .NegAFF2          0.096    0.010    9.498    0.000
   .NegAFF3          0.067    0.009    7.369    0.000
    Agency           1.000
    Intrinsic        1.000
    Extrinsic        1.000
    Positive         1.000
    Negative         1.000
```

Note: The variances of Agency, Intrinsic, etc, are 1.0 because we selected std.lv = TRUE

Outline

Estimation methods

The `estimator` argument in lavaan functions determines the method of estimation

- ML: Maximum likelihood of the complete cases (listwise-delete missings)
  - If `missing = "fiml"` , then Full Information Maximum Likelihood is used to avoid listwise deletion
- WLSMV: weighted least squares with a mean and variance adjustment
  - Standard error estimates that are "robust" to violations of multivariate normality

Bootstrap estimates are also included

- The robust estimators of the standard errors are approximate, but they are widely used.
- lavaan functions include a bootstrap argument
- Why? In small samples, or when parameter distributions are unknown, this is a popular method to evaluate uncertainty.
- The bootstrap estimator will draw repeated random samples and re-estimate the model.

summary

- The `summary()` method in lavaan, as we demonstrated above, generates estimate tables and some summary information.
- Observant readers might have noticed that the default output, obtained from

  ```
  summary(cfa1)
  ```

  did not include the "fit measures" that are often offered with CFA and SEM output in other software.
- We have several ways to deal with that. We can obtain the fit measures by separate requests, or we can insert the request into the summary call.

summary with additional details

- First, we will insert a request for **fit measures** and the **rsquare** into the summary output. The result looks quite a bit more like Mplus. We select ci=FALSE to reduce the width of the output

```
summary(cfa3, rsquare = TRUE, fit.measures = TRUE, ci
    = FALSE)
```

```
lavaan (0.5-22) converged normally after  59 iterations

  Number of observations                           380

  Estimator                                         ML
  Minimum Function Test Statistic              106.847
  Degrees of freedom                                80
  P-value (Chi-square)                           0.024

Model test baseline model:

  Minimum Function Test Statistic             3749.411
  Degrees of freedom                               105
  P-value                                        0.000
```

summary with additional details ...

```
User model versus baseline model:

  Comparative Fit Index (CFI)                    0.993
  Tucker-Lewis Index (TLI)                       0.990

Loglikelihood and Information Criteria:

  Loglikelihood user model (H0)              -3699.110
  Loglikelihood unrestricted model (H1)      -3645.686

  Number of free parameters                         40
  Akaike (AIC)                                7478.219
  Bayesian (BIC)                              7635.826
  Sample-size adjusted Bayesian (BIC)         7508.914

Root Mean Square Error of Approximation:

  RMSEA                                          0.030
  90 Percent Confidence Interval          0.011  0.044
  P-value RMSEA <= 0.05                          0.994

Standardized Root Mean Square Residual:

  SRMR                                           0.031
```

summary with additional details ...

```
Parameter Estimates:

  Information                                    Expected
  Standard Errors                                Standard

Latent Variables:
                   Estimate  Std.Err  z-value  P(>|z|)
  Agency =~
    Agency1           0.466    0.021   22.560    0.000
    Agency2           0.492    0.021   23.774    0.000
    Agency3           0.497    0.022   22.815    0.000
  Intrinsic =~
    Intrin1           0.541    0.039   13.960    0.000
    Intrin2           0.581    0.043   13.489    0.000
    Intrin3           0.615    0.038   16.201    0.000
  Extrinsic =~
    Extrin1           0.388    0.022   17.606    0.000
    Extrin2           0.456    0.027   17.168    0.000
    Extrin3           0.471    0.026   18.100    0.000
  Positive =~
    PosAFF1           0.569    0.028   20.132    0.000
    PosAFF2           0.603    0.029   21.150    0.000
    PosAFF3           0.632    0.029   21.648    0.000
  Negative =~
    NegAFF1           0.634    0.029   21.670    0.000
```

summary with additional details ...

```
    NegAFF2          0.585    0.027   21.457    0.000
    NegAFF3          0.598    0.026   22.805    0.000

Covariances:
                   Estimate  Std.Err  z-value  P(>|z|)
  Agency ~~
    Intrinsic        0.507    0.047   10.782    0.000
    Extrinsic        0.270    0.054    5.028    0.000
    Positive         0.302    0.051    5.950    0.000
    Negative         0.017    0.055    0.312    0.755
  Intrinsic ~~
    Extrinsic       -0.028    0.063   -0.441    0.659
    Positive         0.414    0.052    7.918    0.000
    Negative         0.021    0.060    0.341    0.733
  Extrinsic ~~
    Positive        -0.029    0.058   -0.503    0.615
    Negative         0.209    0.056    3.764    0.000
  Positive ~~
    Negative        -0.071    0.056   -1.274    0.203

Variances:
                   Estimate  Std.Err  z-value  P(>|z|)
   .Agency1          0.048    0.005    9.649    0.000
   .Agency2          0.036    0.005    7.604    0.000
   .Agency3          0.051    0.005    9.278    0.000
```

summary with additional details ...

```
   .Intrin1          0.298    0.029   10.204    0.000
   .Intrin2          0.386    0.036   10.620    0.000
   .Intrin3          0.214    0.029    7.474    0.000
   .Extrin1          0.080    0.009    8.924    0.000
   .Extrin2          0.123    0.013    9.465    0.000
   .Extrin3          0.103    0.012    8.264    0.000
   .PosAFF1          0.121    0.012    9.921    0.000
   .PosAFF2          0.104    0.012    8.615    0.000
   .PosAFF3          0.099    0.013    7.870    0.000
   .NegAFF1          0.107    0.012    9.202    0.000
   .NegAFF2          0.096    0.010    9.498    0.000
   .NegAFF3          0.067    0.009    7.369    0.000
    Agency           1.000
    Intrinsic        1.000
    Extrinsic        1.000
    Positive         1.000
    Negative         1.000

R-Square:
                  Estimate
    Agency1          0.819
    Agency2          0.871
    Agency3          0.830
    Intrin1          0.495
    Intrin2          0.467
```

summary with additional details ...

```
Intrin3       0.639
Extrin1       0.653
Extrin2       0.628
Extrin3       0.682
PosAFF1       0.728
PosAFF2       0.778
PosAFF3       0.802
NegAFF1       0.790
NegAFF2       0.780
NegAFF3       0.842
```

- What other details could we ask for? We checked the lavaan source code, where we find the summary function allows these flags:
  - header (default: TRUE)
  - fit.measures (default: FALSE)
  - estimates (default: TRUE)
  - ci (default: FALSE)
  - fmi (default: FALSE)
  - standardized (default: FALSE)
  - rsquare (default: FALSE)

summary with additional details ...

- std.nox (default: FALSE)
- modindices (default: FALSE)

parameterEstimates

- The "behind the scenes" work to build `summary` output is done by separate functions which can be accessed directly.
- `parameterEstimates()` is doing most of the actual work in the `summary()` display. It can obtain many, or just a few, elements.
- The relevant arguments for `parameterEstimates()` are
  - se (default: TRUE) Show standard errors?
  - zstat (default: TRUE) Show Z, ratio of estimate to standard error
  - pvalue (default: TRUE) Show p value
  - ci (default: TRUE) Show confidence interval
  - level (default: 0.95) Confidence level required to calculate ci
  - boot.ci.type: (default: "perc") Confidence interval for bootstrapped models
  - standardized (default: FALSE) Add standardized parameter estimates
  - fmi (default: FALSE) Show fraction of missing information, for "FIML" models

parameterEstimates ...

- remove.system.eq (default: TRUE) Hide user-constrained parameters
- remove.eq (default.eq: TRUE) Hide system-generated constraints
- remove.ineq (default: TRUE) Hide inequality constraints
- remove.def (default: FALSE) Hide parameter definitions
- rsquare (default: FALSE) Add rows for the R-square

- The default output was too wide for these slides, so we don't look at p or ci. Here are the first 15 lines:

```
parameterEstimates(cfa3, pvalue = FALSE, ci =
    FALSE)
```

parameterEstimates ...

```
         lhs op     rhs   est    se      z
1     Agency =~ Agency1 0.466 0.021 22.560
2     Agency =~ Agency2 0.492 0.021 23.774
3     Agency =~ Agency3 0.497 0.022 22.815
4  Intrinsic =~ Intrin1 0.541 0.039 13.960
5  Intrinsic =~ Intrin2 0.581 0.043 13.489
6  Intrinsic =~ Intrin3 0.615 0.038 16.201
7  Extrinsic =~ Extrin1 0.388 0.022 17.606
8  Extrinsic =~ Extrin2 0.456 0.027 17.168
9  Extrinsic =~ Extrin3 0.471 0.026 18.100
10  Positive =~ PosAFF1 0.569 0.028 20.132
11  Positive =~ PosAFF2 0.603 0.029 21.150
12  Positive =~ PosAFF3 0.632 0.029 21.648
13  Negative =~ NegAFF1 0.634 0.029 21.670
14  Negative =~ NegAFF2 0.585 0.027 21.457
15  Negative =~ NegAFF3 0.598 0.026 22.805
```

parTable

- The function `parTable()` can create a data.frame object that holds the estimated values, with one row per parameter:

```
cfa3.df <- parTable(cfa3)
head(cfa3.df, 10)
```

```
   id       lhs op     rhs user group free ustart exo label
1   1    Agency =~ Agency1    1     1    1     NA   0
2   2    Agency =~ Agency2    1     1    2     NA   0
3   3    Agency =~ Agency3    1     1    3     NA   0
4   4 Intrinsic =~ Intrin1    1     1    4     NA   0
5   5 Intrinsic =~ Intrin2    1     1    5     NA   0
6   6 Intrinsic =~ Intrin3    1     1    6     NA   0
7   7 Extrinsic =~ Extrin1    1     1    7     NA   0
8   8 Extrinsic =~ Extrin2    1     1    8     NA   0
9   9 Extrinsic =~ Extrin3    1     1    9     NA   0
10 10  Positive =~ PosAFF1    1     1   10     NA   0
   plabel start   est    se
1    .p1.     1 0.466 0.021
2    .p2.     1 0.492 0.021
3    .p3.     1 0.497 0.022
4    .p4.     1 0.541 0.039
5    .p5.     1 0.581 0.043
6    .p6.     1 0.615 0.038
7    .p7.     1 0.388 0.022
```

fitMeasures

- A comprehensive list of fit indicators is returned by `fitMeasures(cfa1)`

```
fitMeasures(cfa1)
```

```
          npar             fmin            chisq
        40.000            0.141          106.847
            df           pvalue   baseline.chisq
        80.000            0.024         3749.411
   baseline.df  baseline.pvalue              cfi
       105.000            0.000            0.993
           tli             nnfi              rfi
         0.990            0.990            0.963
           nfi             pnfi              ifi
         0.972            0.740            0.993
           rni             logl unrestricted.logl
         0.993        -3699.110        -3645.686
           aic              bic           ntotal
      7478.219         7635.826          380.000
          bic2            rmsea   rmsea.ci.lower
      7508.914            0.030            0.011
rmsea.ci.upper     rmsea.pvalue              rmr
```

fitMeasures ...

```
           0.044              0.994              0.014
      rmr_nomean               srmr        srmr_bentler
           0.014              0.031              0.031
srmr_bentler_nomean     srmr_bollen  srmr_bollen_nomean
           0.031              0.031              0.031
      srmr_mplus  srmr_mplus_nomean               cn_05
           0.031              0.031            363.332
           cn_01                gfi                agfi
         400.494              0.964              0.946
            pgfi                mfi                ecvi
           0.643              0.965              0.492
```

- Just a few measures? How about the CFI and RMSEA only?

```
fitMeasures(cfa1, fit.measures = c("cfi",
    "rmsea"))
```

```
  cfi rmsea
0.993 0.030
```

fitMeasures ...

- fitMeasures, you will see the list of all possible fit.measures is not fully documented, but it *at least* includes:

```
"cfi","tli","nnfi","pnfi","rfi","nfi",
"ifi","rmsea","rmsea.ci.lower","rmsea.ci.upper",
"rmsea.pvalue","rmr", "srmr", "wrmr","agfi", "pgfi", "mfi", "ecvi",
"baseline.chisq", "baseline.pvalue", "baseline.df"
```

CFA Commentary

- The model appears to fit well,
  - all factor loadings are significant, and the
  - standardized factor loadings indicate strong correlations between indicators and constructs.
- However, we did not explicitly model the relationships among the latent variables.
  - We don't have "Agency" as predictor of "Positive" affect.
  - The CFA fits "unstructured covariances" between latent variables, not "directional regression relationships".

Other follow-up functions

- Other standard R accessor functions are available in lavaan

```
coef(sem1)
fitted(sem1)
resid(sem1)
anova(sem1)
```

- If you run those things, you will notice some wrinkles.
  - Notice that predicted observations (and residuals) are not 1-per-person
  - `anova()` returns a chi-squared test that indicates there are 0 degrees of freedom (that's an SEM concept).

A nice output table

```
library(kutils)
semTable(cfa3, fit = c("cfi", "rmsea"))
```

Parameter	Estimate	SE	z	p
	Factor Loadings			
Agency				
Agency1	0.47	0.02	22.56	.000
Agency2	0.49	0.02	23.77	.000
Agency3	0.50	0.02	22.81	.000
Intrinsic				
Intrin1	0.54	0.04	13.96	.000
Intrin2	0.58	0.04	13.49	.000
Intrin3	0.62	0.04	16.20	.000
Extrinsic				
Extrin1	0.39	0.02	17.61	.000
Extrin2	0.46	0.03	17.17	.000
Extrin3	0.47	0.03	18.10	.000
Positive				
PosAFF1	0.57	0.03	20.13	.000
PosAFF2	0.60	0.03	21.15	.000
PosAFF3	0.63	0.03	21.65	.000
Negative				
NegAFF1	0.63	0.03	21.67	.000

NegAFF2	0.59	0.03	21.46	.000
NegAFF3	0.60	0.03	22.81	.000
	Variances			
Agency1	0.05	0.00	9.65	.000
Agency2	0.04	0.00	7.60	.000
Agency3	0.05	0.01	9.28	.000
Intrin1	0.30	0.03	10.20	.000
Intrin2	0.39	0.04	10.62	.000
Intrin3	0.21	0.03	7.47	.000
Extrin1	0.08	0.01	8.92	.000
Extrin2	0.12	0.01	9.46	.000
Extrin3	0.10	0.01	8.26	.000
PosAFF1	0.12	0.01	9.92	.000
PosAFF2	0.10	0.01	8.62	.000
PosAFF3	0.10	0.01	7.87	.000
NegAFF1	0.11	0.01	9.20	.000
NegAFF2	0.10	0.01	9.50	.000
NegAFF3	0.07	0.01	7.37	.000
	Latent Variances/Covariances			
Agency with Agency	1.00*	0.00		
Intrinsic with Intrinsic	1.00*	0.00		

Extrinsic with Extrinsic	1.00*	0.00		
Positive with Positive	1.00*	0.00		
Negative with Negative	1.00*	0.00		
Agency with Intrinsic	0.51	0.05	10.78	.000
Agency with Extrinsic	0.27	0.05	5.03	.000
Agency with Positive	0.30	0.05	5.95	.000
Agency with Negative	0.02	0.05	0.31	.755
Intrinsic with Extrinsic	-0.03	0.06	-0.44	.659
Intrinsic with Positive	0.41	0.05	7.92	.000
Intrinsic with Negative	0.02	0.06	0.34	.733
Extrinsic with Positive	-0.03	0.06	-0.50	.615
Extrinsic with Negative	0.21	0.06	3.76	.000
Positive with Negative	-0.07	0.06	-1.27	.203

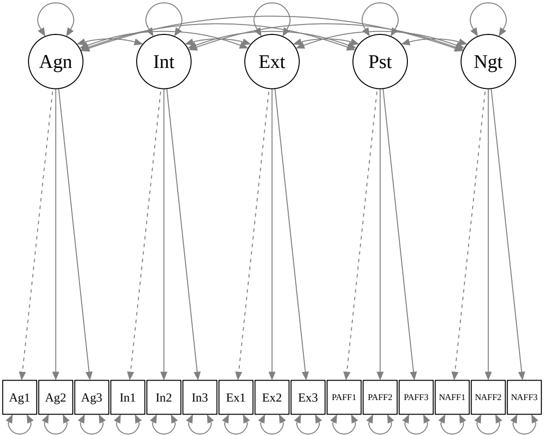
Note. * Indicates parameters fixed for model identification. CFI = 0.99; RMSEA = 0.03.

Plots

- There is no plot method for a lavaan object.
- But there are other packages devoted to creating graphics for models that involve path diagrams (e.g., SEM and neural networks).
- The path diagram is visualized here with the `semPaths()` function in semPlot.

```
library(semPlot)
semPaths(cfa1, layout = "tree2")
```

Plots ...


Agn
Int
Ext
Pst
Ngt
Ag1
Ag2
Ag3
In1
In2
In3
Ex1
Ex2
Ex3
PAFF1
PAFF2
PAFF3
NAFF1
NAFF2
NAFF3

Moderator effects

- Moderators are categorical predictors.
- In a regression context, suppose $Agency_i$ is a continuous predictor of $Positive_i$.

$$Positive_i = \beta_0 + \beta_1 Agency_i, \epsilon_i \sim N(0, \sigma_\epsilon^2) \quad (1)$$

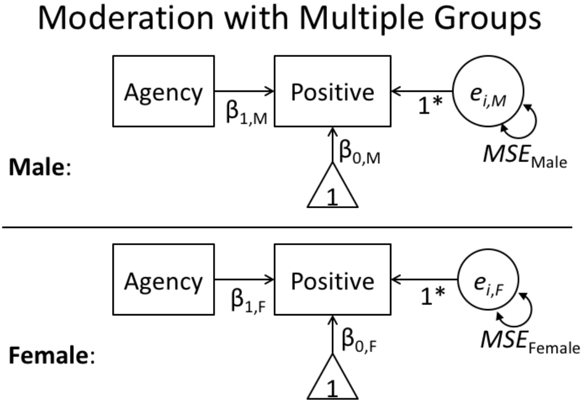
- But we wonder if a categorical variable, $female_i$, causes a change in both the intercept and the slope:

$$Positive_i = \beta_0 + \beta_1 Agency_i + \beta_2 Female_i + \beta_3 Agency_i \cdot Female_i, \epsilon_i \sim N(0, \sigma_\epsilon^2) \quad (2)$$

- Here, the gender is a "**moderator**" of the agency effect

SEM view of the moderator variable

- This sketch segregates the 2 genders entirely; we are estimating 2 separate sets of coefficients.



Outline

2 fits

- Moderation analysis focuses on the differences between groups. In this case, males and females.
- When one of the predictors in an interaction is categorical (e.g., gender), the sem "measurement invariance" approach would lead us to compare
  - a model in which the coefficients for the two groups are assumed to be entirely different, against
  - a simpler model in which some coefficients might be the same

lavaan syntax

```
moderate.mod1 <- 'PosAFF1 ~ c(a, b)*Agency1'
out.mod1 <- sem(moderate.mod1, data = dat, group
    = "Sex")
summary(out.mod1)
```

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

  Number of observations per group
  1                                               195
  2                                               185

  Estimator                                        ML
  Minimum Function Test Statistic               0.000
  Degrees of freedom                                0
  Minimum Function Value               0.000000000000

Chi-square for each group:

  1                                             0.000
  2                                             0.000

Parameter Estimates:
```

lavaan syntax ...

```
  Information                                    Expected
  Standard Errors                                Standard

Group 1 [1]:

Regressions:
                   Estimate  Std.Err  z-value  P(>|z|)
  PosAFF1 ~
    Agency1    (a)    0.326    0.085    3.862    0.000

Intercepts:
                   Estimate  Std.Err  z-value  P(>|z|)
   .PosAFF1           2.317    0.212   10.909    0.000

Variances:
                   Estimate  Std.Err  z-value  P(>|z|)
   .PosAFF1           0.370    0.037    9.874    0.000


Group 2 [2]:

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

lavaan syntax ...

```
  PosAFF1 ~
    Agency1      (b)     0.214     0.099     2.161     0.031

Intercepts:
                      Estimate   Std.Err   z-value   P(>|z|)
   .PosAFF1              2.634     0.245    10.737     0.000

Variances:
                      Estimate   Std.Err   z-value   P(>|z|)
   .PosAFF1              0.481     0.050     9.618     0.000
```

lavaan syntax

```
moderate.mod2 <- 'PosAFF1 ~ c(a, a)*Agency1'
out.mod2 <- sem(moderate.mod2, data = dat, group
    = "Sex")
summary(out.mod2)
```

```
lavaan (0.5-22) converged normally after  17 iterations

  Number of observations per group
  1                                                195
  2                                                185

  Estimator                                         ML
  Minimum Function Test Statistic                0.747
  Degrees of freedom                                 1
  P-value (Chi-square)                           0.388

Chi-square for each group:

  1                                              0.315
  2                                              0.432

Parameter Estimates:
```

lavaan syntax ...

```
  Information                                          Expected
  Standard Errors                                      Standard


Group 1 [1]:

Regressions:
                       Estimate  Std.Err  z-value  P(>|z|)
  PosAFF1 ~
    Agency1     (a)       0.279    0.064    4.336    0.000

Intercepts:
                       Estimate  Std.Err  z-value  P(>|z|)
   .PosAFF1               2.434    0.164   14.829    0.000

Variances:
                       Estimate  Std.Err  z-value  P(>|z|)
   .PosAFF1               0.370    0.038    9.874    0.000


Group 2 [2]:

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

lavaan syntax ...

```
 PosAFF1 ~
   Agency1     (a)     0.279     0.064     4.336     0.000

Intercepts:
                    Estimate   Std.Err   z-value   P(>|z|)
  .PosAFF1             2.476     0.164    15.087     0.000

Variances:
                    Estimate   Std.Err   z-value   P(>|z|)
  .PosAFF1             0.482     0.050     9.618     0.000
```

lavaan syntax

```
anova(out.mod1, out.mod2)
```

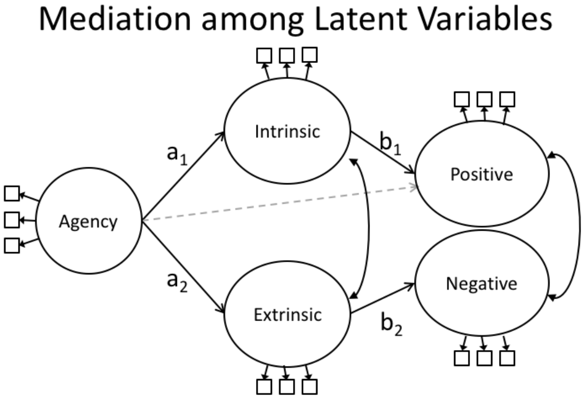
```
Chi Square Difference Test

          Df    AIC    BIC  Chisq Chisq diff Df diff Pr(>Chisq)
out.mod1   0 1335.3 1358.9 0.0000
out.mod2   1 1334.0 1353.7 0.7467    0.74668       1     0.3875
```

Regression relationships among latent variables

- This is the SEM!
- Assume
  1. there are "latent variables" (unmeasured personal traits)
  2. assume that there is a multivariate normal relationship among those traits
  3. the observed scores are a reflection of each individual's latent variables.

The Big Picture

The circles are unmeasured variables.

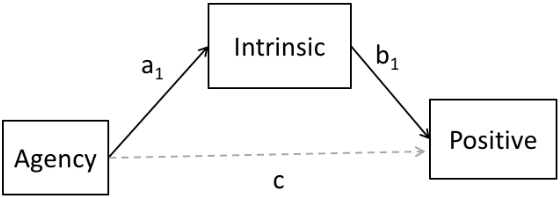


- The SEM integrates

The Big Picture ...

- "measurement error" analysis with
- analysis of the underlying relationships.

The Little Picture



lavaan model will include named parameters

- agency affects intrinsic motivation

$$Intrinsic = a_1 Agency_i + \epsilon 1_i \tag{3}$$

- positiveAffect is affected by intrinsic motivation and agency

$$Positive = b_1 Intrinsic + c\, Agency_i + \epsilon 2_i \tag{4}$$

- The product "$a_1 \cdot b_1$" is the indirect effect of agency on posAffect
- See why? Insert equation (3) into (4):

$$\begin{aligned} Positive &= b_1\{a_1 Agency_i + error_i\} + c\, Agency_i + \epsilon 2_i \qquad (5) \\ &= \{b_1 \cdot a_1\} Agency_i + c\, Agency_i + \{\epsilon 1_i + \epsilon 2_i\} \end{aligned}$$

- Does the "truth" include just the direct effect c or also the indirect effect $a_1 \cdot b_1$?

The Big lavaan formula

```
mediat.mod.2 <- '
 ## measurement model
  Agency =~ Agency1 + Agency2 + Agency3
  Intrinsic =~ Intrin1 + Intrin2 + Intrin3
  Extrinsic =~ Extrin1 + Extrin2 + Extrin3
  Positive =~ PosAFF1 + PosAFF2 + PosAFF3
  Negative =~ NegAFF1 + NegAFF2 + NegAFF3
 ## structural model
  Positive ~ b1*Intrinsic + Agency
  Negative ~ b2*Extrinsic
  Intrinsic ~ a1*Agency
  Extrinsic ~ a2*Agency
  Intrinsic ~~ Extrinsic
 ## define mediation parameters (indirect effects)
  ind1 := a1 * b1
  ind2 := a2 * b2
```

The Big lavaan formula ...

```
'
```

Outline

1. Named parameters for indirect effect $ind1 = a_1 \cdot b_1$

- The product $a_1 \cdot b_1$is the indirect effect of Agency on Positive
- Insert equation (3) into (4):

$$
\begin{aligned}
Positive &= const1 + b_1\{const1 + a_1 Agency_i + error_i\} + c\,Agency_i + \epsilon 2_i \qquad (6\\
&= \{const2 + b_1 const1\} + \{b_1 \cdot a_1\}Agency_i + c\,Agency_i + \{\epsilon 1_i + \epsilon 2_i
\end{aligned}
$$

- Does the "truth" include just the direct effect c or also the indirect effect $a_1 b_1$?

2. Bootstrap the estimate of $a_1 b_1$

- The indirect effect is $a_1 b_1$, a *product*
- The hull hypothesis we would like to test,

$$H_0 : a_1 b_1 = 0$$

- At the current time, no analytical "formula" exists for testing that. Instead, it is necessary to use 'bootstrapped standard errors'.
- The disadvantage of the bootstrap method is that the model must be calculated 100s or 1000s of times (slow!)

Outline

Structural Regressions among Latent Constructs

- The arguments for the sem that we focus on are `se` and `bootstrap`.
- Rather than analytically approximate standard errors, the standard deviation of the bootstrapped estimates is used.

```
Nboot <- 500
mediat.out.2 <- sem(mediat.mod.2, data = dat,
                    std.lv = TRUE, se = "boot",
                      bootstrap = Nboot)
summary(mediat.out.2, standardized = TRUE, fit =
    TRUE)
```

Structural Regressions among Latent Constructs ...

```
lavaan (0.5-22) converged normally after  62 iterations

  Number of observations                           380

  Estimator                                         ML
  Minimum Function Test Statistic              109.152
  Degrees of freedom                                83
  P-value (Chi-square)                           0.029

Model test baseline model:

  Minimum Function Test Statistic             3749.411
  Degrees of freedom                               105
  P-value                                        0.000

User model versus baseline model:

  Comparative Fit Index (CFI)                    0.993
  Tucker-Lewis Index (TLI)                       0.991

Loglikelihood and Information Criteria:

  Loglikelihood user model (H0)              -3700.262
  Loglikelihood unrestricted model (H1)      -3645.686
```

Structural Regressions among Latent Constructs ...

```
  Number of free parameters                           37
  Akaike (AIC)                                  7474.524
  Bayesian (BIC)                                7620.310
  Sample-size adjusted Bayesian (BIC)           7502.917

Root Mean Square Error of Approximation:

  RMSEA                                            0.029
  90 Percent Confidence Interval          0.010    0.043
  P-value RMSEA <= 0.05                            0.996

Standardized Root Mean Square Residual:

  SRMR                                             0.034

Parameter Estimates:

  Information                                   Observed
  Standard Errors                              Bootstrap
  Number of requested bootstrap draws                500
  Number of successful bootstrap draws               500

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

Structural Regressions among Latent Constructs ...

```
  Agency =~
    Agency1          0.466    0.022   20.907    0.000    0.466
    Agency2          0.492    0.023   21.210    0.000    0.492
    Agency3          0.497    0.025   20.179    0.000    0.497
  Intrinsic =~
    Intrin1          0.466    0.037   12.633    0.000    0.540
    Intrin2          0.500    0.035   14.197    0.000    0.580
    Intrin3          0.531    0.033   16.311    0.000    0.616
  Extrinsic =~
    Extrin1          0.373    0.023   16.473    0.000    0.388
    Extrin2          0.440    0.029   15.297    0.000    0.457
    Extrin3          0.454    0.031   14.749    0.000    0.471
  Positive =~
    PosAFF1          0.514    0.026   20.012    0.000    0.569
    PosAFF2          0.545    0.031   17.760    0.000    0.604
    PosAFF3          0.570    0.028   20.217    0.000    0.632
  Negative =~
    NegAFF1          0.620    0.040   15.609    0.000    0.633
    NegAFF2          0.573    0.040   14.298    0.000    0.585
    NegAFF3          0.586    0.033   17.668    0.000    0.598
  Std.all

    0.905
    0.933
    0.912
```

Structural Regressions among Latent Constructs ...

```
    0.703
    0.682
    0.800

    0.808
    0.793
    0.827

    0.854
    0.882
    0.895

    0.889
    0.883
    0.918

Regressions:
                     Estimate  Std.Err  z-value  P(>|z|)   Std.lv
  Positive ~
    Intrinsic (b1)      0.343    0.075    4.560    0.000    0.359
    Agency              0.129    0.076    1.689    0.091    0.116
  Negative ~
    Extrinsic (b2)      0.199    0.066    3.001    0.003    0.202
  Intrinsic ~
```

Structural Regressions among Latent Constructs ...

```
    Agency      (a1)     0.588     0.083     7.083     0.000     0.507
  Extrinsic ~
    Agency      (a2)     0.278     0.065     4.290     0.000     0.268
  Std.all

    0.359
    0.116

    0.202

    0.507

    0.268

Covariances:
                      Estimate   Std.Err   z-value   P(>|z|)    Std.lv
 .Intrinsic ~~
   .Extrinsic           -0.202     0.059    -3.448     0.001    -0.202
 .Positive ~~
   .Negative            -0.076     0.071    -1.082     0.279    -0.076
  Std.all

   -0.202

   -0.076
```

Structural Regressions among Latent Constructs ...

```
Variances:
                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv
   .Agency1           0.048    0.005    9.027    0.000    0.048
   .Agency2           0.036    0.005    7.774    0.000    0.036
   .Agency3           0.050    0.006    8.722    0.000    0.050
   .Intrin1           0.299    0.031    9.737    0.000    0.299
   .Intrin2           0.387    0.039   10.057    0.000    0.387
   .Intrin3           0.213    0.034    6.329    0.000    0.213
   .Extrin1           0.080    0.011    7.127    0.000    0.080
   .Extrin2           0.123    0.015    8.069    0.000    0.123
   .Extrin3           0.103    0.017    6.120    0.000    0.103
   .PosAFF1           0.121    0.015    7.821    0.000    0.121
   .PosAFF2           0.104    0.015    6.748    0.000    0.104
   .PosAFF3           0.099    0.015    6.671    0.000    0.099
   .NegAFF1           0.107    0.016    6.693    0.000    0.107
   .NegAFF2           0.096    0.013    7.434    0.000    0.096
   .NegAFF3           0.067    0.011    5.968    0.000    0.067
    Agency            1.000                               1.000
   .Intrinsic         1.000                               0.743
   .Extrinsic         1.000                               0.928
   .Positive          1.000                               0.815
   .Negative          1.000                               0.959
  Std.all
   0.181
```

Structural Regressions among Latent Constructs ...

```
    0.129
    0.169
    0.506
    0.535
    0.359
    0.348
    0.372
    0.317
    0.271
    0.222
    0.199
    0.210
    0.219
    0.158
    1.000
    0.743
    0.928
    0.815
    0.959

Defined Parameters:
                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv
    ind1              0.202    0.046    4.358    0.000    0.182
    ind2              0.055    0.022    2.502    0.012    0.054
  Std.all
```

Structural Regressions among Latent Constructs ...

```
    0.182
    0.054
```

Outline

Use anova

- The anova() function is a generic in R, it is used in many contexts.
- In SEM, it is used to compare models, to conduct an assessment of the extent to which a simpler model fits the data as well as a more detailed model.
- The mediation model with latent variables estimates fewer parameters than the CFA, but it fits just as well as the CFA.

```
anova(cfa3, mediat.out.2)
```

```
Chi Square Difference Test

             Df    AIC    BIC  Chisq Chisq diff Df diff
cfa3         80 7478.2 7635.8 106.85
mediat.out.2 83 7474.5 7620.3 109.15     2.3051       3
             Pr(>Chisq)
cfa3
mediat.out.2     0.5116
```

Use anova ...

- Intrinsic indicators have the most measurement error (lowest standardized factor loadings). This usually implies more uncertainty about the effect of Intrinsic on other latent variables.

```
summary(mediat.out.2, standardized = TRUE)
```

```
lavaan (0.5-22) converged normally after  62 iterations

  Number of observations                           380

  Estimator                                         ML
  Minimum Function Test Statistic              109.152
  Degrees of freedom                                83
  P-value (Chi-square)                           0.029

Parameter Estimates:

  Information                                 Observed
  Standard Errors                            Bootstrap
  Number of requested bootstrap draws              500
  Number of successful bootstrap draws             500
```

Use anova ...

```
Latent Variables:
                        Estimate  Std.Err  z-value  P(>|z|)   Std.lv
  Agency =~
    Agency1                0.466    0.022   20.907    0.000    0.466
    Agency2                0.492    0.023   21.210    0.000    0.492
    Agency3                0.497    0.025   20.179    0.000    0.497
  Intrinsic =~
    Intrin1                0.466    0.037   12.633    0.000    0.540
    Intrin2                0.500    0.035   14.197    0.000    0.580
    Intrin3                0.531    0.033   16.311    0.000    0.616
  Extrinsic =~
    Extrin1                0.373    0.023   16.473    0.000    0.388
    Extrin2                0.440    0.029   15.297    0.000    0.457
    Extrin3                0.454    0.031   14.749    0.000    0.471
  Positive =~
    PosAFF1                0.514    0.026   20.012    0.000    0.569
    PosAFF2                0.545    0.031   17.760    0.000    0.604
    PosAFF3                0.570    0.028   20.217    0.000    0.632
  Negative =~
    NegAFF1                0.620    0.040   15.609    0.000    0.633
    NegAFF2                0.573    0.040   14.298    0.000    0.585
    NegAFF3                0.586    0.033   17.668    0.000    0.598
  Std.all

    0.905
```

Use anova ...

```
    0.933
    0.912

    0.703
    0.682
    0.800

    0.808
    0.793
    0.827

    0.854
    0.882
    0.895

    0.889
    0.883
    0.918

Regressions:
                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv
  Positive ~
    Intrinsic (b1)    0.343    0.075    4.560    0.000    0.359
    Agency            0.129    0.076    1.689    0.091    0.116
  Negative ~
```

Use anova ...

```
    Extrinsic (b2)     0.199     0.066     3.001     0.003     0.202
  Intrinsic ~
    Agency    (a1)     0.588     0.083     7.083     0.000     0.507
  Extrinsic ~
    Agency    (a2)     0.278     0.065     4.290     0.000     0.268
  Std.all

    0.359
    0.116

    0.202

    0.507

    0.268

Covariances:
                     Estimate   Std.Err   z-value   P(>|z|)    Std.lv
 .Intrinsic ~~
   .Extrinsic          -0.202     0.059    -3.448     0.001    -0.202
 .Positive ~~
   .Negative           -0.076     0.071    -1.082     0.279    -0.076
  Std.all

   -0.202
```

Use anova ...

```
   -0.076

Variances:
                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv
   .Agency1            0.048    0.005    9.027    0.000    0.048
   .Agency2            0.036    0.005    7.774    0.000    0.036
   .Agency3            0.050    0.006    8.722    0.000    0.050
   .Intrin1            0.299    0.031    9.737    0.000    0.299
   .Intrin2            0.387    0.039   10.057    0.000    0.387
   .Intrin3            0.213    0.034    6.329    0.000    0.213
   .Extrin1            0.080    0.011    7.127    0.000    0.080
   .Extrin2            0.123    0.015    8.069    0.000    0.123
   .Extrin3            0.103    0.017    6.120    0.000    0.103
   .PosAFF1            0.121    0.015    7.821    0.000    0.121
   .PosAFF2            0.104    0.015    6.748    0.000    0.104
   .PosAFF3            0.099    0.015    6.671    0.000    0.099
   .NegAFF1            0.107    0.016    6.693    0.000    0.107
   .NegAFF2            0.096    0.013    7.434    0.000    0.096
   .NegAFF3            0.067    0.011    5.968    0.000    0.067
    Agency             1.000                               1.000
   .Intrinsic          1.000                               0.743
   .Extrinsic          1.000                               0.928
   .Positive           1.000                               0.815
   .Negative           1.000                               0.959
```

Use anova ...

```
 Std.all
   0.181
   0.129
   0.169
   0.506
   0.535
   0.359
   0.348
   0.372
   0.317
   0.271
   0.222
   0.199
   0.210
   0.219
   0.158
   1.000
   0.743
   0.928
   0.815
   0.959

Defined Parameters:
                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv
    ind1               0.202    0.046    4.358    0.000    0.182
```

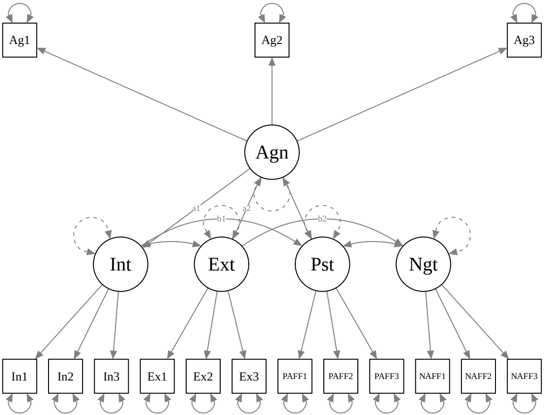
Use anova ...

```
   ind2                 0.055    0.022    2.502    0.012    0.054
  Std.all
   0.182
   0.054
```

plot

```
semPaths(mediat.out.2)
```

plot ...


Ag1
Ag2
Ag3
Agn
a1
a2
b1
b2
Int
Ext
Pst
Ngt
In1
In2
In3
Ex1
Ex2
Ex3
PAFF1
PAFF2
PAFF3
NAFF1
NAFF2
NAFF3

Structural Equation Modeling

- In Psychology, SEM has been an area of tremendous growth since 1980.
- SEM is being absorbed slowly into other fields
- The "gold standard" software for SEM modeling is Mplus, although lavaan has succeeded in "matching" side-by-side many of the calculations.

Session

```
sessionInfo()
```

```
R version 3.3.3 (2017-03-06)
Platform: x86_64-pc-linux-gnu (64-bit)
Running under: Ubuntu 17.04

locale:
 [1] LC_CTYPE=en_US.UTF-8          LC_NUMERIC=C
 [3] LC_TIME=en_US.UTF-8           LC_COLLATE=en_US.UTF-8
 [5] LC_MONETARY=en_US.UTF-8       LC_MESSAGES=en_US.UTF-8
 [7] LC_PAPER=en_US.UTF-8          LC_NAME=C
 [9] LC_ADDRESS=C                  LC_TELEPHONE=C
[11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C

attached base packages:
[1] stats     graphics  grDevices utils     datasets  base

other attached packages:
[1] semPlot_1.1   kutils_1.10   lavaan_0.5-22

loaded via a namespace (and not attached):
 [1] splines_3.3.3          ellipse_0.3-8
 [3] gtools_3.5.0           network_1.13.0
```

Session ...

```
 [5] Formula_1.2-1          assertthat_0.1
 [7] semTools_0.4-14        stats4_3.3.3
 [9] latticeExtra_0.6-28    d3Network_0.5.2.1
[11] lisrelToR_0.1.4        pbivnorm_0.6.0
[13] backports_1.0.5        lattice_0.20-35
[15] quantreg_5.29          quadprog_1.5-5
[17] digest_0.6.12          RColorBrewer_1.1-2
[19] checkmate_1.8.2        ggm_2.3
[21] minqa_1.2.4            colorspace_1.3-2
[23] htmltools_0.3.5        Matrix_1.2-8
[25] plyr_1.8.4             psych_1.6.12
[27] XML_3.98-1.5           SparseM_1.74
[29] xtable_1.8-2           corpcor_1.6.9
[31] scales_0.4.1           whisker_0.3-2
[33] glasso_1.8             sna_2.4
[35] jpeg_0.1-8             openxlsx_4.0.0
[37] fdrtool_1.2.15         lme4_1.1-12
[39] MatrixModels_0.4-1     huge_1.2.7
[41] arm_1.9-3              htmlTable_1.9
[43] tibble_1.2             rockchalk_1.8.101
[45] mgcv_1.8-16            car_2.1-4
[47] ggplot2_2.2.1          nnet_7.3-12
[49] lazyeval_0.2.0         pbkrtest_0.4-6
[51] mnormt_1.5-5           survival_2.40-1
[53] magrittr_1.5           statnet.common_3.3.0
```

Session ...

```
[55] methods_3.3.3          nlme_3.1-131
[57] MASS_7.3-45            foreign_0.8-67
[59] OpenMx_2.7.11          tools_3.3.3
[61] data.table_1.10.4      stringr_1.1.0
[63] munsell_0.4.3          cluster_2.0.5
[65] sem_3.1-9              grid_3.3.3
[67] nloptr_1.0.4           rjson_0.2.15
[69] htmlwidgets_0.8        igraph_1.0.1
[71] base64enc_0.1-3        boot_1.3-18
[73] mi_1.0                 gtable_0.2.0
[75] abind_1.4-5            reshape2_1.4.2
[77] qgraph_1.4.3           gridExtra_2.2.1
[79] knitr_1.15.1           Hmisc_4.0-2
[81] stringi_1.1.2          matrixcalc_1.0-3
[83] parallel_3.3.3         Rcpp_0.12.9
[85] rpart_4.1-10           acepack_1.4.1
[87] png_0.1-7              coda_0.19-1
```