Richard Gonzalez
Psych 614
Version 3.3 (March 2024)

LECTURE NOTES #13: Reliability, Structural Equation Modeling, and Hierarchical Modeling

```
And I have felt
A presence that disturbs me with the joy
Of elevated thoughts; a sense sublime
Of something far more deeply interfused,

(William Wordsworth, Lines Composed a Few Miles Above Tintern Abbey)
```

1. Introduction

   These lecture notes present basic intuitions underlying structural equations modeling (SEM). If you find this technique useful in your research, I suggest that you take a semester-long course on SEM and/or hierarchical linear model (HLM). It turns out that SEM and HLM are essentially the same technique but some analytic problems are easier to implement in one as opposed to the other, so disciplines tend to favor one over the other. I'll give a general overview in these notes but the overview won't replace the content of a semester-long course.

2. Elementary Covariance Algebra

   It will be useful to learn how to work with expectations and variances. I introduce some rules for how to manipulate expectation, variance and covariance. These rules will be useful as we develop intuition for structural equations modeling. The rules are called *covariance algebra* even though they also include operations on means.

covariance algebra rules

   In the following, E refers to expectation, var refers to variance (sometimes just V), and cov refers to covariance (sometimes just C).

   (a) If a is some constant number, then

$$E(a) = a$$

"The average of a constant is a constant." Think about a column in a data matrix. If all the numbers in that column are identical (e.g., all 7s), then the mean will be the value of that number (7).

(b) If a is some constant value real number and X is a random variable with expectation $E(X)$, then

$$E(aX) = aE(X)$$

Multiplying by a constant then averaging is the same as averaging the original data first and then multiplying the average by the constant.

(c) If a is a constant real number and X is a random variable, then

$$E(X + a) = E(X) + a$$

Adding a constant then averaging is the same as averaging the original data first and then adding the constant.

(d) If X is a random variable with expectation $E(X)$, and Y is a random variable with expectation $E(Y)$, then

$$E(X + Y) = E(X) + E(Y)$$

The sum of two averages equals the average of a sum of the two random variables.

(e) Given some finite number of random variables, the expectation of the sum of those variables is the sum of their individual expectations. Thus,

$$E(X + Y + Z) = E(X) + E(Y) + E(Z)$$

(f) If a is a constant real number, and if X is a random variable with expectation $E(X)$ and variance var(X), then the random variable $(X+a)$ has variance var(X). In symbols,

$$\text{var(X)} = \text{var(X + a)}$$

Adding a constant to a column of data does not change the variance.

(g) If a is a constant real number, and if X is a random variable with variance var(X), the variance of the random variable aX is

$$\text{var}(aX) \quad = \quad a^2\text{var(X)}$$

The variance of a column of data that has been multiplied by a constant is equal to variance of the original variable multiplied by the constant squared (a^2).

(h) If X and Y are independent random variables, with variances var(X) and var(Y), respectively, then the variance of the sum $X + Y$ is

$$\text{var(X+Y)} \quad = \quad \text{var(X)} + \text{var(Y)}$$

Similarly, the variance of $X - Y$ when both are independent is

$$\text{var(X-Y)} \quad = \quad \text{var(X)} + \text{var(Y)}$$

(i) Given random variable X with expectation $E(X)$ and the random variable Y with expectation $E(Y)$, then if X and Y are independent,

$$E(XY) \quad = \quad E(X)E(Y)$$

The implication does not go in the other direction. If $E(XY) = E(X)E(Y)$, that does not imply that the two variables are independent.

(j) If $E(XY) \neq E(X)E(Y)$, the variables X and Y are not independent.

This statement is simply the contrapositive of the previous rule.

(k) Given the random variable X with expectation $E(X)$ and the random variable Y with expectation $E(Y)$, then the covariance of X and Y is

$$\text{cov}(X, Y) \quad = \quad E(XY) - E(X)E(Y)$$

(l) Another definition of covariance is

$$\text{cov}(X, Y) \quad = \quad E[(X - \mu_X)(Y - \mu_Y)]$$

(m) If a is a constant and X is a random variable, then the covariance is

$$\text{cov}(X, a) = 0$$

The covariance of a random variable with a constant is zero.

(n) If a is a constant and the variables X and Y are random, then

$$\text{cov}(aX, Y) = a\text{cov}(X, Y)$$

(o) The variance of a random variable X is equal to the covariance of X with itself, i.e.,

$$\text{cov}(X, X) = \text{var}(X)$$

(p) The sum operation between random variables is distributive for covariances , i.e.,

$$\text{cov}(X, Y + Z) = \text{cov}(X, Y) + \text{cov}(X, Z)$$

(q) Now that we know about covariances, we can return to the definition of the sum of two random variables. I will now relax the restriction of independence. If X and Y are random variables, with variances var(X) and var(Y), respectively, then the variance of the sum $X + Y$ is

$$\text{var(X+Y)} = \text{var(X)} + \text{var(Y)} + 2\text{cov}(X, Y)$$

Similarly, the variance of $X - Y$ is

$$\text{var(X-Y)} = \text{var(X)} + \text{var(Y)} - 2\text{cov}(X, Y)$$

(r) Definition of the *correlation*: Given two random variables X and Y, then the covariance divided by the square root of the product of the variances of each variable is the correlation ρ,

$$\rho(X, Y) = \frac{\text{cov}(X, Y)}{\sqrt{\text{var(X)var(Y)}}}$$

The correlation is simply a normalized covariance (i.e., normalized by the product of the standard deviations).

The following results are useful for interactions:

(s) The expectation of the product of two variables is given by the following (using E for expectation and C for covariance):

$$E(XY) = C(X,Y) + E(X)E(Y)$$

(t) The variance of the product of two variables that are normally distributed is (using E for expectation, C for cov and V for var)

$$\begin{aligned} V(XY) = {} & E(X)^2V(Y) + E(Y)^2V(X) + \\ & 2E(X)E(Y)C(X,Y) + \\ & V(X)V(Y) + C(X,Y)^2 \end{aligned}$$

(u) The covariance of two products with all variables distributed multivariate normal is

$$\begin{aligned} C(XY,UV) = {} & E(X)E(U)C(Y,V) + \\ & E(X)E(V)C(Y,U) + \\ & E(Y)E(U)C(X,V) + \\ & E(Y)E(V)C(X,U) + \\ & C(X,U)C(Y,V) + C(X,V)C(Y,U) \end{aligned}$$

3. Covariance Matrices

Suppose you have three variables. There will be three variances (one for each variable) and three covariances (one for each possible pairwise combination). Making sense of these six numbers is facilitated if they are arranged in a matrix such as

	variable 1	variable 2	variable 3
variable 1	var(1)	cov(1,2)	cov(1,3)
variable 2	cov(2,1)	var(2)	cov(2,3)
variable 3	cov(3,1)	cov(3,2)	var(3)

Because cov(x,y)=cov(y,x) (i.e., covariances are symmetric) the lower triangle of the above matrix mirrors the upper triangle[1]. This property is called symmetry and a matrix that has this property is called symmetric.

[1]Triangles are formed on either side of the main diagonal (which starts at the upper left and moves to the lower right).

4. Correlation Matrices

Because a correlation is defined as

$$\text{cor(x,y)} = \frac{\text{cov(x,y)}}{\sqrt{\text{var(x)var(y)}}} \tag{13-1}$$

all information necessary to compute a correlation matrix is present in the covariance matrix.

The elements along the diagonal of a correlation matrix will equal one. The correlation matrix is also symmetric. While it is easy to convert a covariance matrix into a correlation matrix, the conversion will not be possible going the other way (correlation matrix to covariance matrix) unless the variances of each variable are known.

A convenient way to represent both the covariance matrix and the correlation matrix is to display one matrix that has variances in the diagonal, covariances in the upper triangle, and correlations in the lower triangle. That is,

	variable 1	variable 2	variable 3
variable 1	var(1)	cov(1,2)	cov(1,3)
variable 2	cor(2,1)	var(2)	cov(2,3)
variable 3	cor(3,1)	cor(3,2)	var(3)

Such a combined matrix is only for display (usually in published papers) and not for computation.

As we have seen already, these matrices are the building blocks of most multivariate statistical techniques including principal components, factor analysis, discriminant analysis, multivariate analysis of variance, canonical correlation, and structural equations modeling. Some problems are easier to deal with in the form of a covariance matrix and other problems are easier to deal with in the form of a correlation matrix.

5. Using these covariance rules to create new statistical tests: testing two dependent variances

How do you test two variances on variables that come from the same subjects (i.e. are dependent)? This is analogous to the paired-t test for means but applied to variances instead. This test is not easy to find in textbooks or computer programs, but it is straightforward to derive now that we know the covariance algebra rules.

The correlation between the sum and the difference of the two variables is identical

to the difference of the two dependent variances. That is,

$$\text{cov}(X - Y, X + Y) = \text{var}(X) - \text{var}(Y)$$

To see this, take the covariance term on the left hand side, apply the distributive rule and apply the definition that the covariance of a variable with itself equals the variance. The difference of the two dependent variances automatically falls out. One gets an equation like this by working backwards from the desired term (like the difference between two dependent variances) and seeing if there is a way to get that term through another way.

So, testing whether the covariance (or the correlation) between these two scores (sum and difference) is zero is equivalent to testing whether the difference in the dependent variances of X and Y is equal to 0. In other words, to test whether two dependent variances are equal, we can simply test if the correlation between the sum and difference of X and Y—cor(sum, diff)—is different from zero; see LN#6 for testing a correlation against the null hypothesis of 0 through a t-test. It is the same test as presented in LN6, but on the sum and difference as the two variables that are being correlated.

6. Making a connection between a correlation matrix and multiple regression

It is instructive to see how a correlation matrix enters in techniques you already know. Here we will show the connection between a correlation matrix and regression. The "beta coefficients" for each variable in a regression can be computed from the correlation matrix of the variables involved. The "beta coefficients" are the regression weights when all variables (predictors and criterion) are standardized. In such a model there is no intercept. The following discussion is adapted from Edwards (1985).

Let R be the correlation matrix of only the predictors, b be the vector of "standardized beta coefficients"[2], and r is the vector of correlations between each predictor and the criterion. Multiple regression satisfies the following equation:

$$\text{Rb} = \text{r} \tag{13-2}$$

How one multiplies matrices is not important for our purposes (though see the appendix in LN11 for a refresher). All you need to know now is that by "multiplying" R and b we can compute r, the correlation of the criterion with each predictor. But, usually we know r and R and not b; the task is to estimate b. So, think back to middle school. The unknown is b so "divide" both sides of the equation by R to solve for b. Division is computationally a little more difficult for matrices, but the intuition is the same.

[2]For our purposes a vector can be treated as a matrix with one column—these vectors are, not surprisingly, called column vectors.

The correlation matrix R contains the information of how the predictors correlate with each other and the vector r constrains the information of how each predictor correlates with the dependent variable. We observe both matrix R and vector r but we need to estimate vector b (the standardized beta coefficients) from the data.

Back to the regression problem. Look at Equation 13-2—you'll see that all we need to do is multiply both sides by the inverse of R (recall that the product of a matrix with its inverse is equal to the identity matrix).

$$\begin{aligned} \mathrm{Rb} &= \mathrm{r} && (13\text{-}3) \\ \mathrm{R}^{-1}\mathrm{Rb} &= \mathrm{R}^{-1}\mathrm{r} && (13\text{-}4) \\ \mathrm{b} &= \mathrm{R}^{-1}\mathrm{r} && (13\text{-}5) \end{aligned}$$

This shows at all we need to compute the standardized beta coefficients from the regression are the correlations between predictors (matrix R) and the correlations between each predictor and the dependent variable (vector r)

Further, with knowledge of the variances we can transform the "standardized beta coefficients" computed through this method to "raw score regression coefficients", i.e., the regression coefficients that result when the variables are *not* standardized. All one needs to do is multiply the "beta coefficients" by the ratio of the standard deviation of the criterion and the standard deviation of the predictor associated with the slope. That is,

$$\text{raw score reg coef for } \mathrm{x}_i = \text{"beta" for } \mathrm{x}_i \frac{\mathrm{sd(y)}}{\mathrm{sd}(\mathrm{x}_i)} \qquad (13\text{-}6)$$

This section showed that the variance and correlation matrices contain all the information necessary to compute slopes in a regression. This demonstrates that these constructs (means, variances, and covariances) are fundamental to statistics. In other words, once we have the covariance matrix we no longer need the original data to compute regressions. Of course, having the original data is helpful because we can check for outliers, check assumptions, etc., which we cannot easily do when we only have the means and covariances.

The inverse of the covariance matrix, C^{-1}, has a special meaning: it is related to the matrix of partial correlations. That is, if we convert the inverse of the covariance matrix into a correlation matrix[3], each entry in the matrix represents the partial

[3]The computation involves multiplying each off diagonal by -1 and dividing those off diagonals by the square root of the product of the two relevant diagonals. So, for entry 1,2 in the inverse of the covariance matrix C^{-1}, multiply it by -1 and divide by the square root of the product of the diagonal elements corresponding to variables 1 and 2. The R function cov2cor function does this computation except for the multiplication of the off diagonal by -1.

correlation between the two variables in that row and column "controlling" for all the other variables in the matrix. So, if we have 4 variables, the entry in the cell 1,2 of C^{-1} that has been converted to a correlation matrix is equal to the partial correlation between variables 1 and 2 controlling for the other variables 3 and 4. Lecture Notes #8 explains the meaning of the partial correlation and provides a different but equivalent computational approach using full and reduced R^2s. The partial correlation matrix has received recent attention in the context of graphical network models. It is interesting that regression makes use of the related matrix R^{-1} in the computation of the slopes (i.e., Equation 13-5).

```
# check on this claim
library(ppcor)
# toy data
y.data <- data.frame(hl = c(7, 15, 19, 15, 21, 22, 57, 15, 20,
    18), disp = c(0, 0.964, 0, 0, 0.921, 0, 0, 1.006, 0, 1.011),
    deg = c(9, 2, 3, 4, 1, 3, 1, 3, 6, 1), BC = c(0.0178, 1.05e-06,
        1.37e-05, 0.00718, 0, 0, 0, 0.00448, 2.1e-06, 0))
# print partial correlation
pcor(y.data)$estimate
```

```
##              hl       disp        deg        BC
## hl    1.0000000 -0.6720863 -0.6161163 0.1148459
## disp -0.6720863  1.0000000 -0.7215522 0.2855420
## deg  -0.6161163 -0.7215522  1.0000000 0.6940953
## BC    0.1148459  0.2855420  0.6940953 1.0000000
```

```
# check the claim in the text for 1,2
solve(cov(y.data))[1:2, 1:2]
```

```
##              hl       disp
## hl   0.0136871  0.2570525
## disp 0.2570525 10.6876475
```

```
# equals output from pcor()
-1 * 0.2570525/sqrt(0.0136871 * 10.6876475)
```

```
## [1] -0.6720863
```

```
# computation using cov2cor
```

```
temp <- cov2cor(solve(cov(y.data)))
# multipley by -1
temp <- temp * -1
# replace diagonal with 1s
diag(temp) <- 1
# same as pcor() output
temp
```

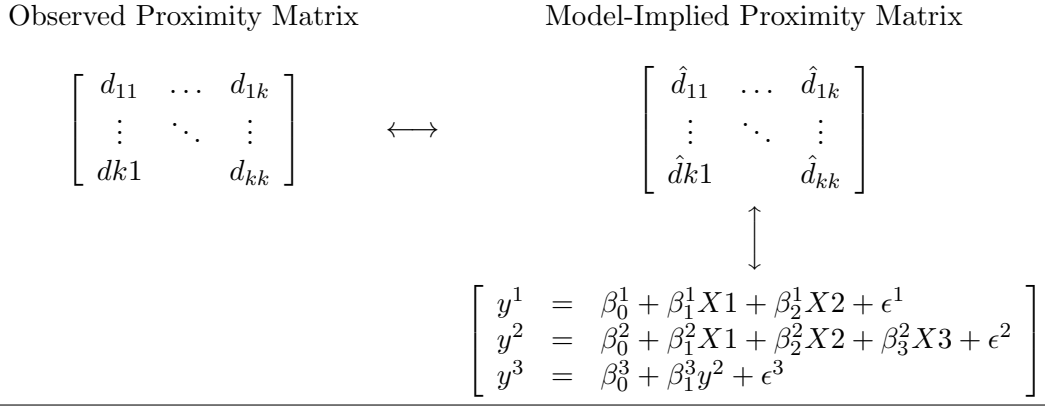
```
##              hl       disp        deg        BC
## hl    1.0000000 -0.6720863 -0.6161163 0.1148459
## disp -0.6720863  1.0000000 -0.7215522 0.2855420
## deg  -0.6161163 -0.7215522  1.0000000 0.6940953
## BC    0.1148459  0.2855420  0.6940953 1.0000000
```

7. The covariance matrix as a similarity matrix

   The covariance matrix is analogous to the "distance matrix" between cities we studied in multidimensional scaling in LN10, but it is a similarity matrix.

   As in MDS (LN10), we have an observed similarity matrix and a model-implied covariance matrix. In the present setting, a set of regression equations that define regression, factor models, growth curves, etc., is what defines the model-implied covariance matrix. We can extend the diagram I introduced in Lecture Notes 10 in MDS to this setting (the new square bracket object is in the lower right, the one with a list of structural models):

Figure 13-1: Relation between Observed Covariance Matrix, Model-Implied Covariance Matrix and a List of Regressions and Various Constraints



A set of regression equations defines a model-implied covariance matrix and we compare the observed covariance matrix to the model-implied covariance matrix. The parameters of the regression equations are found by minimizing the discrepancy between the two covariance matrices. In this hypothetical example, I make use of three dependent variables (the Ys) and three predictor variables (the Xs), such that some predictors apply to some dependent variables and in one case a dependent variable also serves as a predictor of another dependent variable. In this way, we estimate a system of equations. The technique called Structural Equations Modeling (SEM) is a framework to run multiple structural models simultaneously, and this model has PCA, factor analysis, canonical correlation, as well as regression and ANOVA, as special cases. It is possible to test constraints such as equal means or equivalent slopes, say, across several groups or several variables.

We now make use of these concepts to develop intuition to a set of analyses including reliability, mediation and systems of linear equations (aka structural equation modeling). Let's move to reliability.

8. Reliability

What is reliability? The standard definition uses the simple additive model

$$X = T + \epsilon \tag{13-7}$$

where X is the observed score, T is the true score, and ϵ is measurement error. Think of a bathroom scale—you have a true weight T but you observe a reading X that includes both your true weight and any measurement error from the scale. The measurement error could be positive or negative; it behaves in a similar manner to the ϵ in regression/ANOVA but there is a slight difference in interpretation. The ϵ in regression/ANOVA assesses the prediction deviation, that is, the discrepancy between the weighted sum of the predictors and intercept and the observed dependent variable. However, the ϵ in reliability is interpreted as measurement error. We can have both types of error in the same problem. For example, in a regression a predictor might be measured with error (error in predictor) and there is also the discrepancy in prediction (i.e., the usual regression ϵ, which is the difference between the observed dependent variable Y and the predicted Y). We'll come back to this point later in these notes.

The concept of reliability is based on variances. The reliability of variable X is defined as

$$\text{reliability of X} = \frac{\text{var(T)}}{\text{var(T)} + \text{var}(\epsilon)} \tag{13-8}$$

$$= \frac{\text{var(T)}}{\text{var(X)}} \tag{13-9}$$

In words, reliability is defined as a proportion of observed variance that is true vari-

ance. Reliability is interpreted as a proportion—reliability cannot be negative[4].

But in practice we don't know the true score T (nor the error ϵ) so what can we do? How do we separate the true score from the observed score? One approach is a test-retest paradigm[5]. Test the person on the same variable twice. Assuming no changes over time in the true score[6], time 1 is influenced by the same "latent variable" as time 2. The model assumes that the only difference between the two time periods is the error, which I'll denote ϵ_1 and ϵ_2.

So, we have $X_1 = T + \epsilon_1$ and $X_2 = T + \epsilon_2$.[7] We could get a handle on the true score T by examining the covariance between the two observed variables X_1 and X_2. The true component T is what both X_1 and X_2 have in common. We need to assume that the two errors are independent from T and are not correlated with each other (you will see how that assumption simplifies a complicated expression).

$$\begin{aligned} \text{cov}(X_1, X_2) &= \text{cov}(T + \epsilon_1, T + \epsilon_2) \\ &= \text{cov}(T, T) + \text{cov}(T, \epsilon_1) + \text{cov}(T, \epsilon_2) + \text{cov}(\epsilon_1, \epsilon_2) \\ &= \text{cov}(T, T) \text{ under independence} \\ &= \text{var}(T) \end{aligned}$$

Recall that the correlation between two variables is defined as the covariance divided by the product of the standard deviations.

$$\text{cor} = \frac{\text{cov}(X, Y)}{\text{sd}(X)\text{sd}(Y)} \tag{13-10}$$

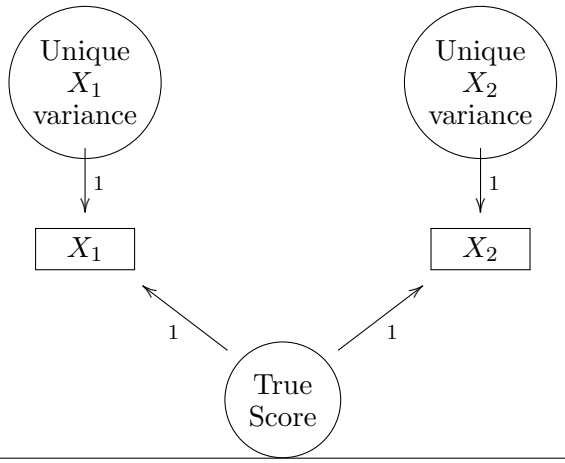
[4]There have been some interesting new insights growing out of connecting traditional reliability definitions to information theory. A recent paper (Markon, 2023) shows that, for normally distributed variables, the traditional definition of reliability is equivalent to the mutual information between the observed score X and the latent score T (known as the Lindley information).

[5]There are many other approaches that I will not cover here.

[6]This is an important assumption. There cannot be change in the intervening time period, so if you expect "growth" or change to occur between the first and second administration, then this definition of reliability is not so useful.

[7]Sometimes the role of true score and noise can flip. For example, in some areas of signal processing the T represents the noise and ϵs represent the signal. If one is recording muscle activity using surface EMG, the electrodes also pick up background noise like the "humming" of the fluorescent lights in the testing room. If there are recordings of two muscle contractions, the hum is the same for both recordings, but the signals may be slightly different due to differences in muscle contraction in the different trials. By subtracting data from recording 1 and recording 2 one can cancel the background hum noise (the part of the signal that is the same at both times) from the part of the signal we care about, which may be different at both times: (hum + sig1) - (hum + sig2) = sig1 - sig2. This contrasts with the true score idea where we correlate in order to "cancel" the noise ϵ and examine the remaining "true" score.

Figure 13-2: Test-retest structural model.



To compute the correlation of X_1 and X_2, we need the covariance over the two Xs (computed above) and each of their standard deviations.

$$\begin{aligned}
\text{cor}(X_1, X_2) &= \frac{\text{cov}(X_1, X_2)}{\sqrt{\text{var}(X_1)\text{var}(X_2)}} \\
&= \frac{\text{var}(T)}{\sqrt{\text{var}(T+\epsilon_1)\text{var}(T+\epsilon_2)}} \\
&= \frac{\text{var}(T)}{\text{var}(T+\epsilon)} \quad \text{assume var of } \epsilon\text{s are equal \& ind} \\
&= \text{reliability}
\end{aligned}$$

Thus, a correlation of a test-retest variable is equivalent to reliability. For this reason, test-retest reliability is usually denoted r_{xx}. This is a case where a raw correlation can be interpreted as a proportion of true variance to observed variance—there is no need to square a correlation in a test-retest situation when it is interpreted as a reliability. In fact, taking the square root of the test-retest reliability leads to an estimate of the correlation between true score and observed score.

This same logic extends to the true correlation between two variables, each measured with their own error. For example, suppose variables X and Y are each measured with error. The correlation between the "true" component of X and the "true" component of Y is equal to:

$$r_{X_T Y_T} = \frac{r_{XY}}{\sqrt{r_{XX} r_{YY}}} \tag{13-11}$$

This equation is important in test theory. It shows that a true correlation is equal to the observed correlation (the numerator in the right hand side) divided by the product of the square roots of the reliability coefficients. So, the product of the reliabilities serves to correct the observed correlation between the two variables. As long as both reliabilities are non-zero and at least one is also not equal to 1, then the correction will always be to increase the value of the observed correlation between X and Y. The correlation in Equation 13-11 is called the **"correction for attenuation."**

A few words on latent variables. Latent variables seem to rub people the wrong way at first. It seems we should only base empirical science on what we can directly observe. Actually, there are many examples in science of latent variables (constructs we don't directly observe). One recent example is the black hole in astronomy. A black hole cannot be observed directly, but the presence of a black hole can be observed indirectly, such as the pattern light makes when it travels near a black hole (e.g., see the link black holes). This serves as an analogy for how latent variables are used in structural equation modeling. One has a set of observations, or indicators, and they jointly determine the unobserved, latent variable. The latent variables can be used to make testable predictions. Latent variables are used throughout science; they are not something limited to social science or statistics or, more specifically, to structural equations modeling and correlations.

9. Spearman-Brown (SB) Formula

For a complete derivation of the SB formula see a good text book on psychometrics such as Nunnally's *Psychometric Theory*.

The logic of the SB formula is to estimate the reliability of a k item test from knowledge of the reliability of individual items. We know that measurement instruments with more items should have higher reliability all things being equal, and the SB formula quantifies that intuition. For example, suppose I have a 5 item questionnaire and I know the intercorrelations of each item with the other items. However, I do not know (without further computation) how a person's score on this 5 item test (i.e., the sum over the 5 items) would correlate with their summed score on another 5 items (where the items are sampled from the same domain), or how the sum of these five items today would correlate with the same five items tomorrow. In other words, we would like to know the reliability of the five item test as a total score, but we only observe the individual reliabilities of each item.

We denote the correlation of an item with another item as r_{11} (some books use r_{ij} to denote item i with item j). The correlation of a sum of k items with another sum of k items is denoted r_{kk}. We know that r_{kk} should be greater than or equal to r_{11}.

Now comes the famous SB formula that relates r_{11}, the reliability of a single item, to r_{kk}, the reliability of a set of k items. To estimate r_{kk} we take

$$r_{kk} = \frac{kr_{11}}{1+(k-1)r_{11}}$$

Thus, from the inter-item correlations between k items we estimate the reliability of the sum of k items. This form is known as the standardized version because it is based on correlations. There is also an unstandardized version that I present below based on covariances rather than correlations. SPSS and R compute both standardized and unstandardized.

The limit of SB as k goes to infinity (an infinitely large test) is 1, which represents perfect reliability. This means that as the number of items in a scale grows, the reliability will increase as well to the upper limit of 1.

To study the behavior of SB we can use a simple R function, which produces Table 1. The columns show different values of k (number of items) and the rows show different values of r11 (item correlation). The entries in each cell are the SB values. As the number of items increases, the SB increases as well, so even with very low item-level reliabilities (.1), the SB reaches .74 when there are 25 items.

```
library(xtable)
SB <- function(r11, k) {
   k*r11/(1+(k-1)*r11)
}

rvals <- c(.1,.2,.3,.4,.5)
kvals <- c(2,3,4,5,10,25)
SBvalues <- matrix(0,length(rvals),length(kvals))
for (i in 1:length(rvals))
  for (j in 1:length(kvals))
    SBvalues[i,j] <- SB(rvals[i],kvals[j])
rownames(SBvalues) <- as.character(rvals)
colnames(SBvalues) <- as.character(kvals)

print(xtable(SBvalues,digits=2, table.placement="!h",
 caption.placement="top", label="tab:sb",
 caption=
 "SB for number of items (cols) and r11 (rows)"))
```

10. Cronbach's α

	2	3	4	5	10	25
0.1	0.18	0.25	0.31	0.36	0.53	0.74
0.2	0.33	0.43	0.50	0.56	0.71	0.86
0.3	0.46	0.56	0.63	0.68	0.81	0.91
0.4	0.57	0.67	0.73	0.77	0.87	0.94
0.5	0.67	0.75	0.80	0.83	0.91	0.96

Table 1: SB for number of items (cols) and r11 (rows)

What do you do if you have many different inter-item correlations, so each item has a different r_{11}? You want to estimate a single r_{kk} using Spearman-Brown, but how do you choose among the different observed r_{11}s?

Nothing deep here. Simply take the average correlation, apply S-B to the average, and you get what's known as the standardized Cronbach's α. If you had thought of that, then everyone would be saying YOURNAMEGOESHERE's α.

So, to estimate r_{11} from k items compute all possible correlations and *average* (yes, average) the correlations to estimate a singe r_{11}. In symbols,

$$r_{11} = \frac{\sum_{ij} r_{ij}}{\left(\begin{array}{c} k \\ 2 \end{array} \right)}$$

where $\left(\begin{array}{c} k \\ 2 \end{array} \right)$ is "k choose 2" (the total number of correlations that went into the average).

Let's do this example from Nunnally & Bernstein. Imagine we have 3 judges who rate 8 subjects. We want to know the reliability of these three judges. One way to frame this question is how much do these three judges intercorrelate (here just take the average inter-item correlation). Another way to frame this question is to ask what is the reliability of the sum of the three judges (i.e., if these judges were to rate another eight subjects and we summed the three judge's scores, what would be the correlation of the original eight sums with the new eight sums?). This latter question is answered by applying the SB formula on the average inter-item correlation (which is called Cronbach's α).

Example from data in Nunnally & Bernstein.

Items	Judges		
1	1	1	1
2	1	1	1
3	0	0	0
4	0	1	0
5	0	0	0
6	0	1	1
7	0	0	0
8	0	1	1

Cronbach's α in SPSS

The SPSS syntax for this problem is

```
data list free / j1 j2 j3.

begin data
1 1 1
1 1 1
0 0 0
0 1 0
0 0 0
0 1 1
0 0 0
0 1 1
end data.

reliabilities  variables = j1 j2 j3
 /statistics all.
```

The intercorrelations of these judges are

	J1	J2	J3
J1	1		
J2	.4472	1	
J3	.5774	.7746	1

If you take the average correlation and apply SB, then you get the standardized alpha. In this example the average correlation r is .5997, apply SB to that and you get .8180. SPSS reports a "standardized item alpha" = .8180, which is the same thing!

Some textbooks report different ways of computing alpha. For example, using the covariance matrix (so this is the nonstandardized form of alpha)

	J1	J2	J3
J1	.2143		
J2	.1071	.2679	
J3	.1429	.2143	.2857

Sum of all elements in the covariance matrix (9 terms) is 1.6965. The sum of the diagonal elements is .7679. Cronbach's alpha is given by (Nunnally & Bernstein 6-26)

$$\alpha = \frac{k}{k-1}\left(1 - \frac{\Sigma\sigma_a^2}{\sigma_y^2}\right) \quad (13\text{-}12)$$

$$= \frac{3}{2}\left(\frac{1.6965 - .7679}{1.6965}\right) \quad (13\text{-}13)$$

$$= .821 \quad (13\text{-}14)$$

This is the unstandardized form of Cronbach's α.

A completely different, but equivalent, way to compute Cronbach's α is through the intraclass correlation and then apply the SB formula. To compute the intraclass, first compute the ANOVA source table using items and judge as two factors (no interaction term is fitted because there is only one observation per cell as in the randomized block design from LN4).

Source table (k=3 judges):

	MS	SS	df
items	.5655	3.958	7
judge	.292	.583	2
residual	.1012	1.417	14
total		5.958	23

The ANOVA intraclass is .6046. This is an index of similarity of one judge to the next; it is not the average inter-judge correlation. The equation for the ANOVA-based intraclass is

$$\text{intraclass} = \frac{\text{MSB-MSW}}{\text{MSB + (k-1)MSW}} \quad (13\text{-}15)$$

$$= \frac{.5655 - .1012}{.5655 + (2).1012} \quad (13\text{-}16)$$

$$= .6046 \quad (13\text{-}17)$$

All the info needed for this computation is included in the ANOVA source table, which decomposes the variances into different components.

If you apply Spearman-Brown to this intraclass

$$r_{kk} = \frac{kr_{11}}{1+(k-1)r_{11}} \quad (13\text{-}18)$$

$$= \frac{3(.6046)}{1+2(.6046)} \quad (13\text{-}19)$$

$$= .821 \quad (13\text{-}20)$$

This last number is interpreted in terms of how this set of 3 judges will correlate to another set of 3. Again, the same number as before, the unstandardized Cronbach's α.

With a little algebra both steps (the intraclass and the SB formula) can be condensed into a single expression:

$$\frac{\text{MSB-MSW}}{\text{MSB}} \quad (13\text{-}21)$$

This is what appears on page 277 Eq 7-20 in Nunnally & Bernstein. Cronbach's α is identical to applying the SB formula to the intraclass correlation, and the simple equation in these notes Eq 13-21 shows that the relevant info is embedded in ANOVA. So ANOVA also shows up in reliability theory[8].

It may seem strange to use the sum of squares (SS) for items rather than SS for judges (aka raters). Judges are conceptualized as a blocking factor, so we want to remove from the error term any effect due to mean differences between the judges. So, suppose the judges perfectly agree with each other. In that case, all three scores for each item will be the same, there won't be any within item variance (so MSW goes to 0) and all the remaining variance is between items (MSB). In this extreme case when MSW = 0, the intraclass will be equal to 1. The other extreme is that all variance is within items (i.e., all three judges always disagree with each other), so MSB=0 and the intraclass in Equation 13-15 goes to -1/(k-1). The intraclass correlation is asymmetric as it only approaches 0 when k is very large.

It is helpful to see how the same concept, Cronbach's α, emerges from different analytic perspectives.

Cronbach's α in R

I'll show three different ways to compute Cronbach alpha in R. One way to compute alpha in R is through the alpha command in the psych package. It prints both standardized and unstandardized (called raw in the output) versions and provides additional information such as a 95% CI around alpha, some descriptive statistics and information on missing data.

[8] There is a subtle difference in whether maximum likelihood estimation (MLE) or restricted maximum likelihood estimation (REML) is used. MLE doesn't use degrees of freedom; we will see that estimation used in the SEM exmaple in a few pages. REML uses degrees of freedom, which is what this ANOVA estimator is doing. More on this later when we introduce SEM.

```r
library(psych)

# define rater data
rater.data <- matrix(c(1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0,
    0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1), ncol = 3, byrow = T)
colnames(rater.data) <- c("j1", "j2", "j3")

# print covariance matrix
round(cov(rater.data), 3)
```

```
##       j1    j2    j3
## j1 0.214 0.107 0.143
## j2 0.107 0.268 0.214
## j3 0.143 0.214 0.286
```

```r
# EXAMPLE of alpha using psych package
alpha(rater.data)
```

```
##
## Reliability analysis
## Call: alpha(x = rater.data)
##
##   raw_alpha std.alpha G6(smc) average_r S/N  ase mean   sd
##       0.82      0.82    0.79       0.6 4.5 0.11 0.46 0.43
##  median_r
##      0.58
##
##     95% confidence boundaries
##          lower alpha upper
## Feldt     0.40  0.82  0.96
## Duhachek  0.61  0.82  1.03
##
##  Reliability if an item is dropped:
##    raw_alpha std.alpha G6(smc) average_r S/N alpha se var.r
## j1      0.87      0.87    0.77      0.77 6.9     0.09    NA
## j2      0.73      0.73    0.58      0.58 2.7     0.19    NA
## j3      0.62      0.62    0.45      0.45 1.6     0.27    NA
##    med.r
## j1  0.77
## j2  0.58
## j3  0.45
```

```
##
##  Item statistics
##    n raw.r std.r r.cor r.drop mean   sd
## j1 8  0.77  0.79  0.60   0.54 0.25 0.46
## j2 8  0.87  0.86  0.80   0.70 0.62 0.52
## j3 8  0.92  0.92  0.89   0.80 0.50 0.53
##
## Non missing response frequency for each item
##       0    1 miss
## j1 0.75 0.25    0
## j2 0.38 0.62    0
## j3 0.50 0.50    0
```

```
# this function prints two significant digits; check same
# as above
alpha(rater.data)[[1]]
```

```
##  raw_alpha std.alpha   G6(smc) average_r      S/N       ase
##  0.8210526 0.8180083 0.7878248 0.5997202 4.494757 0.1091044
##       mean        sd  median_r
##  0.4583333 0.4341567 0.5773503
```

Another way to compute alpha is through an SEM approach using the lavaan package. This requires understanding of SEM, which we will cover later in these notes, but I will go ahead and show it now. This model estimates the factor loadings on a single factor, where those factor loadings are set to be equal to each other, and the variances of the observed variables j are also set equal to each other. This model has 3 variances and 3 covariances as input (the covariance matrix) and estimates two parameters (equal coefficient to create the factor and the factor variance). Think a kind of factor model (from LN11) where the elements of the first eigenvector are equal and the errors (ϵs) are also estimated and set equal to each other for a kind of homogeneity of variance across variables assumption.

The Rsquared from this SEM, where the loadings are forced to be equal and residual variances are forced to be equal, is equivalent to the intraclass correlation. Just apply S-B to this intraclass of .605 and you get Cronbach's alpha.

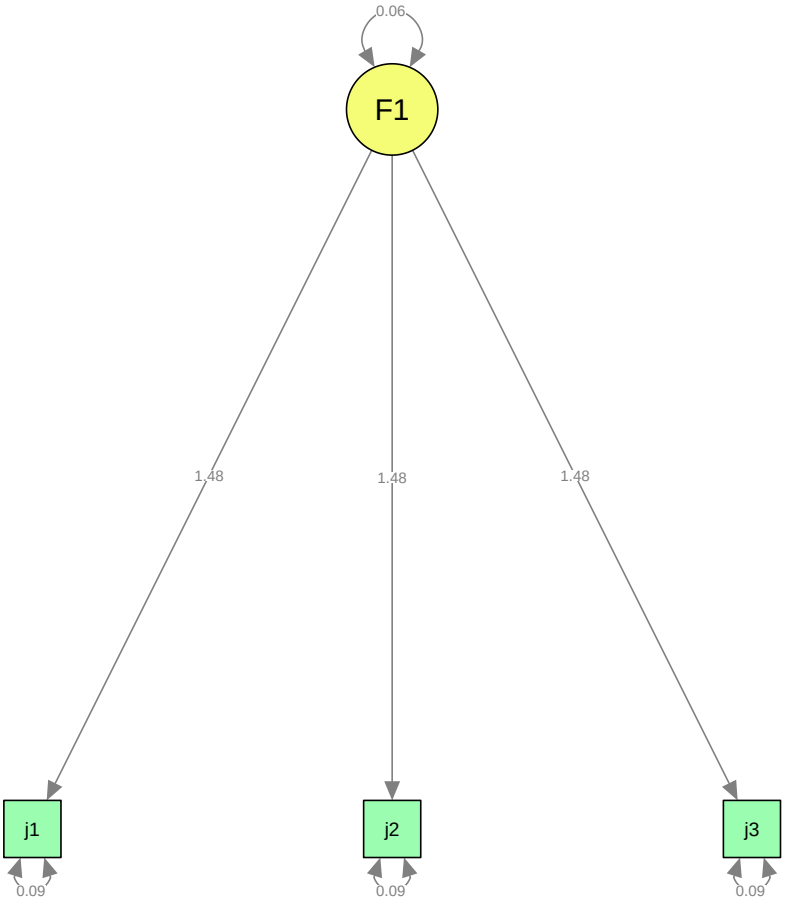
```
#warning message is suppressed
library(lavaan)
library(semPlot)
library(semTools)
```

```
#EXAMPLE of alpha using lavaan (SEM framework)
#factor loading is set to 1 & error variances set
#equal to each other
model.rater <- 'F1 =~ l1*j1 + l1*j2 + l1*j3
j1 ~~ v1*j1
j2 ~~ v1*j2
j3 ~~ v1*j3
'
out.cfa <- cfa(model.rater,data=rater.data,
               auto.fix.first=F)
summary(out.cfa,rsquare=T,standardized=T)
```

```
## lavaan 0.6.17 ended normally after 8 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                         7
##   Number of equality constraints                     4
##
##   Number of observations                             8
##
## Model Test User Model:
##
##   Test statistic                                 2.245
##   Degrees of freedom                                 3
##   P-value (Chi-square)                           0.523
##
## Parameter Estimates:
##
##   Standard errors                             Standard
##   Information                                 Expected
##   Information saturated (h1) model          Structured
##
## Latent Variables:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   F1 =~
##     j1        (l1)    1.477    0.001 1387.519    0.000
##     j2        (l1)    1.477    0.001 1387.519    0.000
##     j3        (l1)    1.477    0.001 1387.519    0.000
##    Std.lv  Std.all
##
##     0.368    0.778
##     0.368    0.778
```

```
##     0.368    0.778
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##    .j1        (v1)    0.089    0.031    2.828    0.005
##    .j2        (v1)    0.089    0.031    2.828    0.005
##    .j3        (v1)    0.089    0.031    2.828    0.005
##     F1                0.062    0.038    1.633    0.102
##    Std.lv  Std.all
##     0.089    0.395
##     0.089    0.395
##     0.089    0.395
##     1.000    1.000
##
## R-Square:
##                    Estimate
##     j1                0.605
##     j2                0.605
##     j3                0.605
```

```
semPaths(out.cfa,
whatLabels="est",color = list(lat = rgb(245, 253, 118, maxColorValue = 255),
    man = rgb(155, 253, 175, maxColorValue = 255)))
```



The R-square printed here is .605, so apply the SB formula and you get Cronbach's α of .82 as we saw before. Or you can make use of the reliabilities command in the SemTools package and get Cronbach's α directly from the output of the earlier lavaan call. The double headed arrows attached to the squares represent the variance of the residuals (i.e., the variance of the "plus epsilons").

```
reliability(out.cfa)
```

```
##                F1
## alpha  0.8210526
## omega  0.8210526
## omega2 0.8210526
## omega3 0.8210526
```

```
## avevar 0.6046512
```

This output also mentions other measures of reliability such as three different forms of a measure called omega (ω). For this model the three omega measures are equivalent to Cronbach's alpha but omega works for more general models that do not impose the equality restrictions on both the loadings and the error variances I used to show alpha. Below is a more general model for which the loadings and the error variances are not assumed to be equal. Note that Cronbach's alpha remains the same because it is estimated relative to the more constrained model we used earlier, but the three omega's are now different from Cronbach's α. For a recent description of omega see Flora (2020) in the journal *Advances in Methods and Practices in Psychological Science*. As we have seen throughout this year, the assumptions you want to make can have major implications on the results of your modeling. I also encourage you to examine books on psychometric theory and reliability theory to cover many developments in those fields such as specific measures of reliability for use in various contexts such as ordinal data.

The notion that reliability theory and measures such as Cronbach's α can be defined in terms of a special kind of factor analysis where one specifies which variables relate to factors and the variables all load equally to the factor is quite interesting. In this sense, reliability can be interpreted as the proportion of variance that is attributable to the common shared component where all variables contribute equally. This differs from usual factor analysis where variables are allowed to contribute differentially to the factor (i.e., have different factor loadings). Here I'll run a traditional factor analysis by letting the three variables load differentially on the factor (each of the three js can have different loadings instead of being constrained to have equal loading, with one loading constrained to 1 to set the scale). This model adds four parameters to estimate because now there are three residual variances to estimate rather than one residual variance (so two more variances) and two loadings rather than one loading (so one more loading for a total of three additional parameters—two more residual variances and one more loadings).

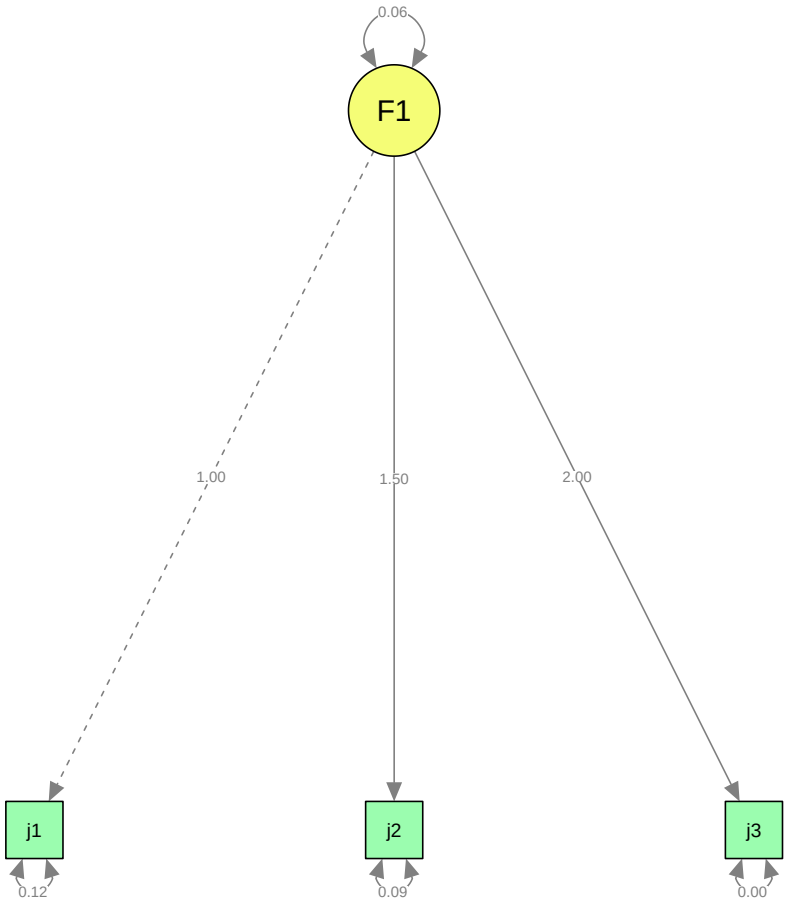
```
model.rater2 <- 'F1 =~ j1 + j2 + j3'
out.cfa.free <- cfa(model.rater2,data=rater.data)
summary(out.cfa.free,rsquare=T,standardized=T)
```

```
## lavaan 0.6.17 ended normally after 22 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                         6
##
```

```
##   Number of observations                                8
##
## Model Test User Model:
##
##   Test statistic                                    0.000
##   Degrees of freedom                                    0
##
## Parameter Estimates:
##
##   Standard errors                             Standard
##   Information                                 Expected
##   Information saturated (h1) model          Structured
##
## Latent Variables:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   F1 =~
##     j1                1.000
##     j2                1.500    0.866    1.732    0.083
##     j3                2.000    1.291    1.549    0.121
##    Std.lv  Std.all
##
##     0.250    0.577
##     0.375    0.775
##     0.500    1.000
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##    .j1                0.125    0.068    1.852    0.064
##    .j2                0.094    0.074    1.265    0.206
##    .j3                0.000    0.102    0.000    1.000
##     F1                0.063    0.074    0.840    0.401
##    Std.lv  Std.all
##     0.125    0.667
##     0.094    0.400
##     0.000    0.000
##     1.000    1.000
##
## R-Square:
##                    Estimate
##     j1                0.333
##     j2                0.600
##     j3                1.000


reliability(out.cfa.free)
```

```
##                F1
## alpha  0.8210526
## omega  0.8526316
## omega2 0.8526316
## omega3 0.8526316
## avevar 0.6744186
```

```
semPaths(out.cfa.free, whatLabels="est",color =
 list(lat = rgb(245, 253, 118, maxColorValue = 255),
 man = rgb(155, 253, 175, maxColorValue = 255)))
```



These two different factor models can be compared to each other; that is, we can test whether the improvement in fit by allowing different loadings is statistically different from the model that forces the loadings to be equal across the raters (in the spirit of a "full minus reduced" test). The model with the constraints is the one that is related to reliability theory, whereas the model with free loadings is a traditional confirmatory factor analysis. The sample size in this example is small so we cannot reject the hypothesis that freeing up the constraints improves the fit.

```
anova(out.cfa.free, out.cfa)
```

```
##
## Chi-Squared Difference Test
##
##              Df    AIC    BIC  Chisq Chisq diff RMSEA
## out.cfa.free  0 33.446 33.923 0.0000
## out.cfa       3 29.692 29.930 2.2454     2.2454     0
##              Df diff Pr(>Chisq)
## out.cfa.free
## out.cfa            3     0.5231
```

I'll cover more of the SEM logic in the remaining pages of these lecture notes. For now, note how similar this is to the factor model approach in LN11, which looks like a single factor model along with error variances (the "plus ϵs"). We'll see soon that SEM also encompasses regression and ANOVA, so it becomes the mother of all statistical methods covered across these two semesters.

The third way to compute the intraclass reliability and SB in R is through a linear mixed model. We analyze the data in long form (remember this from LN5 for repeated measures). Each videotape receives its own random intercept and we are assessing the agreement within the three raters by decomposing the sum of squares in much the same way I showed using ANOVA in the earlier example. In this setting, the intraclass is the ratio of the between variance over the sum of the between variance and the residual variance.

```
library(lme4)

# put data in long format
rater.data.long <- data.frame(rating = c(rater.data), rater = rep(c("j1",
    "j2", "j3"), each = 8), video = factor(rep(1:8, 3)))

# run lmer using maximum likelihood to match the SEM MLE
# output
```

```
out.lmer <- lmer(rating ~ rater + (1 | video), rater.data.long,
    REML = F)
summary(out.lmer)
```

```
## Linear mixed model fit by maximum likelihood  ['lmerMod']
## Formula: rating ~ rater + (1 | video)
##    Data: rater.data.long
##
##      AIC      BIC   logLik deviance df.resid
##     33.7     39.6    -11.8     23.7       19
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -1.41502 -0.58038 -0.02432  0.68540  1.60516
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  video    (Intercept) 0.13542  0.3680
##  Residual             0.08854  0.2976
## Number of obs: 24, groups:  video, 8
##
## Fixed effects:
##             Estimate Std. Error t value
## (Intercept)   0.2500     0.1673   1.494
## raterj2       0.3750     0.1488   2.521
## raterj3       0.2500     0.1488   1.680
##
## Correlation of Fixed Effects:
##         (Intr) ratrj2
## raterj2 -0.445
## raterj3 -0.445  0.500
```

```
# compute the intraclass; i.e., 0.13542/(0.13542+0.08854)
k <- VarCorr(out.lmer)
k$video[1]/(k$video[1] + summary(out.lmer)$sigma^2)
```

```
## [1] 0.6046512
```

This value of the intraclass correlation emerging from the linear mixed model (using MLE) is identical to what we saw earlier in the SEM example (also using MLE),

printed again for easy reference:

```
reliability(out.cfa)
```

```
##               F1
## alpha  0.8210526
## omega  0.8210526
## omega2 0.8210526
## omega3 0.8210526
## avevar 0.6046512
```

Cronbach's alpha will then be for k = 3, k*0.6046512/(1+(k-1)*0.6046512) = 0.8210527.

For completeness, the ANOVA intraclass estimate using REML instead of MLE is

```
# run lmer using REML
out.lmer.reml <- lmer(rating ~ rater + (1 | video), rater.data.long,
    REML = T)
summary(out.lmer.reml)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: rating ~ rater + (1 | video)
##    Data: rater.data.long
##
## REML criterion at convergence: 29.8
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -1.32363 -0.54290 -0.02275  0.64113  1.50149
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  video    (Intercept) 0.1548   0.3934
##  Residual             0.1012   0.3181
## Number of obs: 24, groups:  video, 8
##
## Fixed effects:
##             Estimate Std. Error t value
## (Intercept)   0.2500     0.1789   1.398
## raterj2       0.3750     0.1591   2.358
## raterj3       0.2500     0.1591   1.572
```

```
##
## Correlation of Fixed Effects:
##         (Intr) ratrj2
## raterj2 -0.445
## raterj3 -0.445  0.500
```

```
# compute the intraclass; i.e., 0.1548/(0.1548+0.1012)
k <- VarCorr(out.lmer.reml)
k$video[1]/(k$video[1] + summary(out.lmer.reml)$sigma^2)
```

```
## [1] 0.6046512
```

11. Reliability & Regression with One Predictor

How does measurement error influence a linear regression with one predictor? When the dependent variable has measurement error there is no serious problem because that is exactly the kind of error regression is designed to deal with (but there will be loss of statistical power). For instance, take the usual regression model

$$Y = \beta_0 + \beta_1 X + \epsilon \tag{13-22}$$

Measurement error on the dependent variable

If the dependent Y has measurement error, then the ϵ term in the regression captures both measurement error on Y and the prediction error of the equation. A simple regression cannot distinguish measurement noise in the criterion from noise in the prediction. That is, if instead of writing Y we write "true Y plus noise of Y" on the left hand side

$$Y_T + \epsilon_Y = \beta_0 + \beta_1 X + \epsilon \tag{13-23}$$

$$Y_T = \beta_0 + \beta_1 X + (\epsilon - \epsilon_Y) \tag{13-24}$$

The "true component" of Y (Y_T) is taken to be a linear transformation of the variable X plus the combined measurement error on Y and the usual regression prediction error being lumped into the error term (i.e., the single term in the parentheses).

Measurement error on Y reduces the power of the tests (e.g., increases the size of the confidence interval around β_1), but does not introduce bias. The requirement is that the error be independent from all the effects in the model. Violations of these assumptions can be dealt with easily (e.g., if errors are correlated across observations, as in a time series, then there exist techniques to take into account the "autocorrelation").

However, there is no provision in the standard regression model for measurement error on the predictor X. More generally, there is no provision in the standard regression

model for measurement error on any predictor variable. But frequently regressions involve predictors that contain measurement error (e.g., the total score on the self-esteem scale might be used as a predictor but it has measurement error, the behavioral coding of a rat's licking behavior may have error in measurement, etc). It turns out that measurement error on X not only reduces power, but it also introduces bias. The problems associated with bias are, in general, more serious than reduction in power. Thus, the inability for regression to handle measurement error in the predictor(s) is a very serious limitation. I'll develop some intuition about these limitations and then suggest ways of addressing them.

Error in one predictor

We now turn to an examination of the effects of having a "noisy" predictor in a simple regression equation with just one predictor; this discussion is adapted from Neter, Wasserman, and Kutner (1985). The only machinery we need to reach a better understanding the effects of measurement error in regression is covariance algebra. Consider a simple linear regression as displayed in Equation 13-22. The kind of regression model we want to study is

$$Y = \beta_0 + \beta_1 X_T + \epsilon \tag{13-25}$$

where X_T denotes the true X score. However, when there is measurement error on X (I'm using X to denote the observed score), then the desired model can be expressed as

$$Y = \beta_0 + \beta_1 (X - \epsilon_x) + \epsilon \tag{13-26}$$

where ϵ_x is the measurement error on X. The previous two equations are identical because X_T = X-ϵ_x.

This last equation shows that a crucial assumption in regression is being violated: independence of error terms. By re-arranging terms we get

$$Y = \beta_0 + \beta_1 X + (\epsilon - \beta_1 \epsilon_x) \tag{13-27}$$

and can see that the total error is correlated with X[9]. Major problems arise when the independence assumption is violated.

I have just shown that the independence assumption is violated when there is measurement error on X. Now let's see exactly what happens to the β_1 term in the regression equation when there is measurement error on X. Below I show that the β_1 under measurement error for X will always be less than the true β_1 (i.e., the true beta without measurement error).

[9] As a do-it-yourself exercise, you might want to show the covariance between X and $(\epsilon + \beta_1 \epsilon_x)$ equals $\beta_1 \text{var}(\epsilon_x)$. This value will not usually be zero and thus, there is a violation of the independence assumption because now the residual will be correlated with the predictor.

We need to define some terms. Y is the dependent variable, X is the observed predictor, X_T is the true predictor, ϵ_{X} is the measurement error on X, and ϵ is the measurement error on Y.

The β_1 for the regression of Y on X is equal to

$$\frac{\mathrm{cov}(X,Y)}{\mathrm{var}(X)} \tag{13-28}$$

(this is just the standard definition of the slope). Similarly, the true β_1, denoted β_1^{true}, for the regression of Y on $\mathrm{X_T}$ is

$$\frac{\mathrm{cov}(X_T,Y)}{\mathrm{var}(X_T)} \tag{13-29}$$

The covariance of X and Y is equal to (under independence of ϵ_x from Y)

$$\begin{aligned} \mathrm{cov}(X_T+\epsilon_x,Y) &= \mathrm{cov}(X_T,Y) && (13\text{-}30)\\ &= \beta_1^{\mathrm{true}}\mathrm{var}(X_T) && (13\text{-}31)\end{aligned}$$

One more definition. The reliability, as discussed above, will be denoted r_{xx}. The xx subscript suggests the interpretation of the correlation of X with itself (as in a test-retest situation). Following the definition of reliability (Equation 13-9)

$$r_{xx} = \frac{\mathrm{var}(X_T)}{\mathrm{var}(X)} \tag{13-32}$$

Now the last step, recall that the slope using the observed predictor is

$$\frac{\mathrm{cov}(X,Y)}{\mathrm{var}(X)} \tag{13-33}$$

But I just showed that the cov(X,Y) = $\beta_1^{\mathrm{true}}\mathrm{var}(X_T)$, so

$$\begin{aligned} \beta_1 &= \frac{\mathrm{cov(X,Y)}}{\mathrm{var(X)}} && (13\text{-}34)\\ &= \frac{\beta_1^{\mathrm{true}}\mathrm{var}(X_T)}{\mathrm{var(X)}} && (13\text{-}35)\\ &= \beta_1^{\mathrm{true}} r_{xx} && (13\text{-}36)\end{aligned}$$

Now because r_{xx}, the reliability of X, is always a number between 0 and 1, β_1 will be less than or equal to β_1^{true}. Thus, measurement error introduces systematic bias

on the estimate of the slope parameter. Things get more complicated with multiple predictors as I show below.

Remedial measures for measurment error

What do you do when your predictor has measurement error? There are several ways to deal with measurement error in the predictor. First, if you found significance with a "noisy" predictor, then you are probably okay because noise probably made the test less powerful[10]. The problem comes when you fail to find significance—it might be that the true X variable is related, but the measurement error masked that relationship. Second, a useful technique involves an "instrumental variables." An instrumental variable is a variable that is correlated with the true X but not with the measurement error on X. Instrumental variables can then be included in the analyses in a way that helps adjust for the error issue. A practical problem with this technique is that it is difficult to know when a variable is correlated with true X but not with ϵ_x (i.e., satisfying the definition of an instrumental variable)—often, this argument relies on theory and commonsense but it usually cannot be empirically tested. A third technique involves the use of structural equations models (SEM). These models include measurement error directly in the regression equation. Constructing a model and developing an algorithm for estimating/testing the parameters of the model is a good thing to do for many reasons, including that one can asses and evaluate the adequacy of the model, and hence, the adequacy of the approach to address measurement error. I'll cover both instrumental variables and SEM later in these lecture notes.

12. Reliability and Multiple Regression

All bets are off when trying to get a handle on the effects of measurement error on predictors in the context of multiple regression. Measurement error can increase, decrease, or even change the sign of correlations and betas at the observed level. As you might suspect, getting a deep understanding of the effects of measurement error in the context of multiple regression is not easy. Following Cohen and Cohen (1983, pp. 406-413) we will build intuition by using the notion of partial correlations.

Recall from Lecture Notes 8 that a partial correlation involves a correlation between two variables of interest where the effects of a third variable (typically a nuisance variable or a covariate, but sometimes a theoretically meaningful variable such as a mediator) have been removed. One way to think about this is to imagine a correlation between Y and X where you want to "partial out" the effects of a variable W. The partial correlation between Y and X (partialling the effects of W) is equivalent to performing two regressions (Y on W and X on W) and correlating the residuals from the two regressions. In other words, the regression of Y on W creates residuals that have the linear effect of W "removed" from Y and the regression of X on W creates

[10] This may not generally be the case because the above result was obtained under the assumption that the measurement error on X is independent from Y. How violations of such assumptions influence both β_1 and the variance of β_1 is very complicated and no simple statement can be made.

residuals that have the linear effect of W "removed" from X. The correlation of these two sets of residuals is the partial correlation.

As shown in LN8, a relatively simple way to compute the partial correlation is by the formula

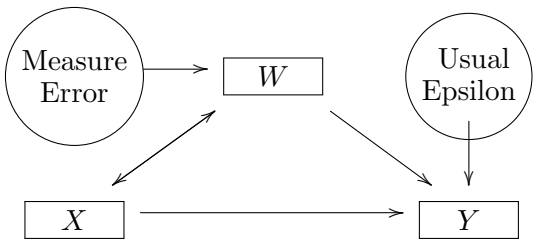
$$r_{yx.w} = \frac{r_{yx} - r_{yw}r_{xw}}{\sqrt{(1 - r_{yw}^2)(1 - r_{xw}^2)}} \quad (13\text{-}37)$$

The partial correlation formula (Equation 13-37) provides a convenient way to understand the effects of measurement error on each of the variables. Cohen and Cohen (1983, p406-413) discuss the different effects of having measurement error on Y, X, and W. Here is a table showing the effects of measurement error on the observed partial correlation. We allow measurement error on variable W and assess the effects on the partial correlation $r_{yx.w}$. I only display a few cases (all for reliability .7), but the effects are more dramatic for lower reliability.

example	r_{yx}	r_{yw}	r_{xw}	r_{ww}	$r_{yx.w}$	true $r_{yx.w}$
1	.3	.5	.6	.7	0	-.23
2	.5	.7	.5	.7	.24	0
3	.5	.7	.6	.7	.14	-.26
4	.5	.3	.8	.7	.45	.57
5	.5	.3	.6	.7	.42	.37

Look at how discrepant the observed partial is from the true partial. That is, with a reliability of .7, which many people take to be acceptable level of reliability in social science. Things get worse with lower reliability, and extremely more complicated with more than two predictor variables and less than perfect reliability on two or more of those predictors. One nice feature of an ANOVA design is that the predictors are known (e.g., you assign subjects to groups) so the predictors have perfect reliability (assuming we didn't make any coding or assignment errors), so an ANOVA design doesn't have to deal with measurement error in the predictors in the same way as the more general regression that can not only use grouping variables as predictors but also measured variables as predictors. Of course, add a measured covariate to the ANOVA and now you have the same problem we see in regression because the measured covariate likely contains measurement error.

The structural model representing this multiple regression with a predictor having error is depicted below.



As before, in the figure squares indicated observed variables and circles indicate unobserved or latent variables. We do not observe measurement error or prediction error directly but estimate it from data. Y has two predictors so two arrows coming into its box. Predictors X and W are allowed to be correlates as with any regression and the correlation is depicted in the graph with a double headed arrow. One of the predictors, W, contains measurement error. You can derive tables like the one below by using the definition of partial correlation and the covariance algebra rules introduced at the beginning of the these lecture notes.

A more general point about measurement error in regression. The above table showing that the observed partial correlation (and hence regression beta) can move around a lot and not necessarily track the true partial correlation makes me wonder about people who make the case that they are using "evidence-based approaches" to justify their arguments. For example, recent arguments about whether or not GREs should be included in graduate school admissions has proponents of both sides making claims they have evidence-based recommendations. typically based on analyses involving multiple regressions. In what I have seen, most of these arguments are flawed because, among many reasons, they don't take into account the measurement error of the variables in their regression equations. They say "Look at the evidence. This is what the regression shows." But, typically, the regressions are not trustworthy because they didn't properly account for measurement error, selection effects, etc. We can't trust the regressions because measurement error can completely change the sign of a regression coefficient as the table above showed. My point is that the material reviewed in this section should make us suspicious of hiding behind the "evidence-based" mantra if that just means we blindly run a regression without deeper thinking. A meta-analysis won't help with this issue either because these measurement error issues (and other issues like selection effects) are sprinkled throughout the literature making it difficult to interpret the aggregate results in a meta-analysis.

See Horn (1963, Acta Psychologica, 21, 184-271) for additional relations between covariance matrices and a discussion of methodological issues such as correlations with difference scores.

13. Instrumental Variables

Instrumental variables analysis is a technique common in several social sciences. It allows one to exploit a randomization-like structure to get better causal estimates even in the context of a correlational design. A simple example, taken from Gennetian et al, 2008, *Developmental Psychology*, 44, 381-, involves the relation between mother's education attainment (ME) and child's performance (CP). At best, the relation between those two variables is a correlation and without additional information or assumptions one could not move toward "causal inferences" of the effect of ME on CP. Instrumental variables serve the role of that additional information allowing one to do a little more with correlational information; random assignment is another way to move toward causal inferences, but random assignment is not always possible.

Suppose that moms are randomly assigned to a treatment condition (T), such as an intervention designed to influence her educational attainment ME (but not influence in a direct way the child's educational performance). Mom's educational attainment (ME) is assessed after this intervention. One can run a regression with ME = β_0 $+\beta_1$T + ϵ_1, which if T is a dummy code, then, as we saw in Lecture Notes 7 showing the connection between regression and the two-sample t test, the slope represents the difference between the treatment and control means. That part of the design establishes the "causal effect" of the treatment T on ME. But we still don't know about the outcome of interest CP. Do changes in ME "cause" subsequent changes in CP? Is it possible to influence CP by influencing ME?

We can make use of some of the structure in this setup and make a few assumptions. We will use what is called a "two stage least squares estimation", which is just a fancy way of saying run one regression, take relevant output from that regression and use that output in a second regression (so two stages[11]). The first regression is the one I mentioned in the previous paragraph: compute the predicted values $\widehat{\text{ME}}$ based on the instrument T. This represents the predicted educational attainment of mom given her random assignment to treatment T; those predicted scores do not include the residuals (i.e., noise) and hence any correlation between the residuals ϵ_1 and other variables have been removed from the predicted score. In other words, ME and CP could be correlated because they share a common variable, but by using the predicted values $\widehat{\text{ME}}$ that are associated with the treatment T we reduce the impact of such issues like common factor and correlated residuals.

Then use that predicted value of ME (denoted $\widehat{\text{ME}}$) in a second equation to predict the child's educational performance, i.e., the regression CP = β'_0 $+\beta'_1\widehat{\text{ME}}$ + ϵ_2 where primes are used to distinguish the slope and intercept from the previous regression. The two epsilons are denoted by subscripts 1 and 2 for first and second regression. If we make some assumptions, such as the correlation between the treatment and the residuals in the second regression is 0, we can derive an estimate of the direct effect

[11]There are now more efficient and better approaches than two-stage least squares but the logic of the procedure is the same. The improvements have been more on the computational side of how the regression weights and standard errors are computed.

(some would say the causal link) between mother's educational attainment and child's performance. This example used a random assignment of the variable T, but econometricians have shown that even when there isn't random assignment of T, as long as some additional assumptions are met, one can use this logic to derive causal estimates even from a correlation design. The logic is that randomization creates a situation where these assumptions are automatically satisfied but in a correlation design one has to make these additional assumptions (which can sometimes be checked).

The two step regression I outlined above yields the right parameter estimate, but the standard error for the regression slope β_1' will be off because the second regression does not properly model the correlation between the epsilons. There are specialized programs, including ones in SPSS and R, that run instrumental variable analyses and yield the proper error term. Here is some example syntax using the variables in the example I've been discussing.

Instrumental variables in SPSS

```
2SLS CP WITH ME
  /INSTRUMENTS T
  /CONSTANT.
```

You specify the DV and critical predictor in the first line and define the instrument in the second line. The output looks like this:

```
Model Summary
Equation 1 Multiple R .432
R Square .187
Adjusted R Square .142
Std. Error of the Estimate 6.206

ANOVA
Sum of Squares df Mean Square F Sig.
Equation 1 Regression 159.454 1 159.454 4.140 .057
Residual 693.291 18 38.516
Total 852.745 19

Coefficients
                        B       SE    beta    t     p
Equation 1 (Constant) 14.597 42.438         .344 .735
M                       .863   .424 1.361 2.035 .057
```

Instrumental variables in R

In R the two stage least squares function is ivreg in the AER package, with the same

results.

```
library(AER)
summary(ivreg(CP ~ ME | T, data = data))
```

```
##
## Call:
## ivreg(formula = CP ~ ME | T, data = data)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -8.856 -4.586 -0.844  2.609 14.290
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  14.5966    42.4379   0.344   0.7349
## ME            0.8629     0.4241   2.035   0.0569 .
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.206 on 18 degrees of freedom
## Multiple R-Squared: -1.586,Adjusted R-squared: -1.729
## Wald test:  4.14 on 1 and 18 DF,  p-value: 0.05688
```

We can use our new knowledge about covariance algebra to figure out what all the fuss is about with instrumental variables. We know that a slope is equal to the ratio of the covariance between the DV and the predictor and the variance of the predictor (LN#6). Thus, in the second regression just described, the slope β_1' can be rewritten as

$$\text{cov}(\text{CP},\widehat{\text{ME}})/\text{var}(\widehat{\text{ME}}) \tag{13-38}$$

Substituting in the regression equation for $\widehat{\text{ME}}$ leads to the familiar equation for the instrumental slope after making some assumptions about correlated error

$$\beta_1' = \frac{\text{cov}(\text{CP, T})}{\text{cov}(\text{ME, T})} \tag{13-39}$$

When the instrument is binary (such as two groups), this slope amounts to a ratio of two differences: the difference in treatment means for the dependent variable CP over the difference in treatment means for the predictor variable mother's educational

attainment. It isn't obvious right now why that is the case, but below I derive this claim using covariance algebra.

This equation can be derived from taking Equation 13-38 and using covariance algebra to re-express the terms to simplify to Equation 13-39. That is,

$$\begin{aligned}
\beta_1' &= \frac{\text{cov}(\text{CP}, \widehat{\text{ME}})}{\text{var}(\widehat{\text{ME}})} \\
&= \frac{\text{cov}(\text{CP}, \beta_0 + \beta_1 T)}{\text{var}(\beta_0 + \beta_1 T)} \\
&= \frac{\beta_1 \text{cov}(\text{CP,T})}{\beta_1^2 \text{var}(\text{T})} \\
&= \frac{\text{cov}(\text{CP,T})}{\text{cov}(\text{ME,T})}
\end{aligned}$$

Note that $\widehat{\text{ME}}$ (predicted ME from the first regression in the two stage model) in this setup does not have a residual variance so the numerator and denominator in the second line don't include an ϵ_1. The last line follows because $\beta_1 = \frac{\text{cov}(\text{ME,T})}{\text{var}(\text{T})}$.

Equivalent equations that you may see in the literature are (1) a ratio of two betas (e.g., ratio of slopes from the regression of CP on T and regression of ME on T) and, (2) if T is a binary variable indicating membership in one of two groups, the IV estimator defined as the ratio of the difference of the means on the DV over the difference of the means on the predictor variable; that is, $\frac{\mu_{Y1} - \mu_{Y2}}{\mu_{P1} - \mu_{P2}}$ where Y is the DV (CP in our running example), P stands for the predictor variable (ME in our running example) and numbers 1,2 refer to groups 1 and 2. You can use covariance algebra rules to show the equivalence of these different expressions for the IV estimator.

I introduce this example here to illustrate how covariance algebra can illuminate some complicated statistical ideas. The instrumental variables approach is not easy to understand, but by using covariance algebra the idea and the importance of its assumptions fall out naturally (e.g., illustrating the role of no correlation between the residuals from the two equations, the instrument is not correlated with the outcome).

Finally, I'll show how to run the two regressions manually so you can see how the resulting β from the instrumental variables procedure comes about. To recap the procedure: run one regression with the instrument predicting the predictor, save the fitted values, and use the fitted values to predict the dependent variable. The betas are the same as ivreg() in R and in SPSS but the standard errors are off; do not trust the standard errors out of lm() when using two-stage least squares. One needs special programs like ivreg() to compute the proper standard errors. Still, this example

illustrates the same betas emerge so we verify our understanding of this approach and the value of covariance algebra.[12]

```
reg1 <- lm(ME ~ T, data = data)
fitted.vals <- predict(reg1)
reg2 <- lm(CP ~ fitted.vals, data = data)
summary(reg2)
```

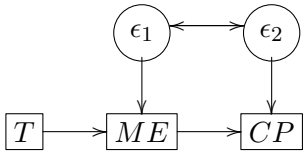
```
##
## Call:
## lm(formula = CP ~ fitted.vals, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.5118 -2.0608  0.0495  1.6410  4.0363
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  14.5966    16.8009   0.869    0.396
## fitted.vals   0.8629     0.1679   5.139 6.87e-05 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.457 on 18 degrees of freedom
## Multiple R-squared:  0.5947,Adjusted R-squared:  0.5722
## F-statistic: 26.41 on 1 and 18 DF,  p-value: 6.872e-05
```

The instrumental variables logic can be applied in a structural equation modeling (SEM) context as well. One needs to be careful because SEMs are usually estimated in a maximum likelihood context whereas instrumental variables are traditionally estimated in a two stage least squares context, so the estimates and their standard errors may differ slightly if one compares the output from SEM to the output from a traditional instrumental variables program.

The SEM representation of the instrumental variables approach is given in this Figure. The underlying logic is that there is correlation between the residuals of the two key variables, ME and CP in this case, and the point of using an instrument is to get around that correlation (i.e., by using the predicted values from using the instrument

[12] One would not get the same betas if residuals are used instead of the fitted values or if both predictor and instrument are included in the same regression with two predictors. I'll show this is in the mediation subsection below.

as a regression, the correlation between the residuals is in some sense broken). The assumption is that the instrument correlates with one of the variables but not the other, so we can use the betas from regressions to control for the possible correlation between the two key variables we can't otherwise control. This is very much the spirit of what random assignment to conditions can accomplish, but in instrumental variables this is conducted statistically. Obviously, a randomization design is preferable but if that cannot be accomplished, then the instrumental variables approach could be used to generate an estimate of the causal effect.



This graph is similar to the mediation model (presented later in these notes) but the focus of the instrumental variables approach is on the intermediate connection between ME and CP. That is, focus is on the slope β_1' in the Figure above and the method gives the slope an interpretation that is less correlational in the sense that the procedure "breaks" a key threat to causal interpretation compared to the mere correlation between those variables). In mediation analysis, the goal is to decouple the direct and indirect paths between T and CP. One of the key assumptions of an instrument is that it NOT have a direct path to the final variable, so the instrument should not directly influence CP except through the "mediator." In the mediation literature this situation of no direct effect is called perfect mediation.

connection between instrumental variables and reliability

An interesting observation is that if one views the instrument as another measurement of the predictor variable (as in test-retest), then the covariance between the instrument and the predictor is the numerator in the test-retest correlation (see earlier in these lecture notes; it is equal to the variance of the "true" score). The beta of the instrumental variable approach is closely related to the disattenuated correlation in test theory, which shows that one can account for the bias in a correlation due to less than perfect reliability through the formula

$$\text{cor(Tx, Ty)} \quad = \quad \frac{\text{cor(x, y)}}{\sqrt{\text{rel(x)*rel(y)}}}$$

where x and y are observed variables, Tx represents "true" x (i.e., x without measurement error) and rel(x) represents the reliability of x (see the relevant definitions and results presented earlier in these lecture notes).

To see this, let T1 and T2 be the X and the instrument, respectively, and Y be the dependent variable. We know that the instrumental variable estimate is cov(Y,T2)/cov(T1,T2).

Because cov(T1,T2) estimates V(T) and cov(Y,T2) is equal to r(Y,T2)sqrt(V(Y)*V(T1)), we can rewrite the instrumental variable estimate as

$$\frac{\text{r(Y,T2)}}{\sqrt{\text{rel(Y) * rel(T2)}}}\sqrt{\frac{\text{V(true Y)}}{\text{V(true T)}}}.$$

This is simply the classic disattenuated correlation from test theory times a factor of the ratio of the square root of the true score variance for Y over the true score variance for T. The factor of the ratio of square root of the true score variances converts the estimate of the true score correlation (a concept from test theory) to the estimate of the true score regression beta (a concept from the instrumental variable approach). This is an unusual interpretation of the instrumental variable approach but it falls out completely from the covariance algebra; it shows that there is a deep connection between classic test theory and modern approaches of instrumental variables. Most proponents of instrumental variables believe they are doing something newer than classic test theory but now you know better (☺). Bollen has run with this idea and explored ways of implementing more general SEM approaches using instrumental variables. Some of this work has been implemented in his R package MIIVsem (I provide an example in the R appendix).

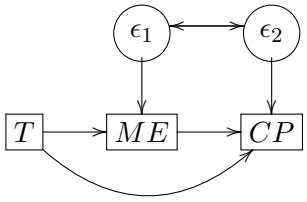
It may appear that a natural use of instrumental variables in behavioral work is when one has a manipulation check and wants to examine the effect of the manipulation check variable on the dependent variable. In this case the variable representing the manipulation is the instrument (e.g., group 1 vs group 2). The manipulation check becomes the predictor of interest such that the researcher wants to examine how the variability in the manipulation played out (i.e., some people responded more to the manipulation than others), so the key question becomes how the manipulation check predicts the dependent variable instead of how the intended manipulation predicts the dependent variable. The latter acts as though everyone responded the same way to the treatment, the former uses the "actual" response to the manipulation as measured by the manipulation check. In other words, if one uses the actual manipulation variable as the only predictor, then one examines, in a sense, the "intent to treat" because the analysis focuses on the condition the subjects are in not how they responded to the treatment. However, if one uses the manipulation variable as an instrument, runs a regression with the manipulation check as the dependent variable, uses the predicted values from that regression as predictors in the second stage regression, makes several assumptions, then it may appear they are examining the effect of the manipulation check (how the manipulation actually played out) on the dependent variable. The problem is that the predicted scores in the first regression are two values (i.e., the $\hat{y}$ for group 1 and the $\hat{y}$ for group 2), so when entered in the second regression as a predictor the t test is essentially identical to the test from the regression using the actual manipulation check variable. In the end, using a manipulation check as the predictor and the actual manipulation values as the instrument leads to similar results (look at page 13-40 for the case of a binary instrument showing a ratio of mean differences, of course, the standard error will be slightly different too). An analogous critique can be made about the practice of using the manipulation check

as a mediator between the actual manipulation and the dependent variable, though that is more subtle because it is the residuals of the mediator that play a key role in determining the path between the mediator and the outcome variable.

The potential use of instrumental variables in correlation studies is compelling. One still needs to make a set of assumptions that cannot easily be empirically validated. The pessimistic take is that if one doesn't use instrumental variables, then in order to interpret the correlation as "causal" one must make lots of hand-wavy arguments (i.e., assumptions) about there not being a third variable, that causation goes in one direction, etc. In order to use instrumental variables, one's hands are waving just as much, but in the end one has a formula that leads to an estimate of the "causal relation." While I recognize that randomized studies cannot always be carried out, I can't break away from my traditional upbringing that "cause" should be reserved for randomized experiments. I can't distinguish whether the pair of waving hands from the person who uses instrumental variables is more or less convincing than the pair of waving hands from the person who doesn't and tries to convince me why the reported correlation should be interpreted in a causal manner. In both cases people are waving their hands. I'm happy using correlations in my research because they provide important information about associations and also using randomized designs when possible to address causality. I try to be mindful about when I can and cannot use causal language in my thinking based on the nature of the data and study design.

14. Mediation

The basic mediation analysis uses a similar structure to instrumental variables but the emphasis is on a different part of the path diagram. Whereas in instrumental variables the focus is on examining the path between M and Y, in mediation analysis the question is whether X influences Y directly (the curved path) and/or whether X influences Y indirectly through M. So, mediation boils down to a question about partitioning possible pathways: a direct path from X to Y, a mediated path through M, both, or neither.



A colleague gave me a nice way to conceptualize the meaning of an indirect path. Suppose you tell me a secret and I turn around and tell your advisor. One wouldn't say that you told your advisor (i.e., there was no direct communication link between you and your advisor). I was the middleman who took input from you and spewed it

to your advisor. I'm the "mediator" in that example—there are two communication links, one from you to me and another from me to your advisor (the direct link between you and your advisor did not occur).

Much of behavioral science research begins along the lines of simple stimulus-response descriptions (questions such as does X influence Y). Those research questions are black box questions because there isn't much interest in understanding how the relation works, rather the interest in merely in describing the relation. A mechanistic or process view focuses on an intermediate process of how X influences Y. If I pay my son to spend more time studying, and then he gets a high grade on a test, the stimulus-response operational language is that payment led to high grade. That is all one needs to understand according to behaviorism. But did the payment directly lead to a higher grade? Does the test "know" how much my son got paid and the higher grade reflects the payment? Most likely there is a more nuanced account. For example, payment for studying led my son to spend more time studying, which meant he learned the material better, which led to better comprehension, which led to a higher performance on the exam. This can be compared to the counterfactual where I didn't pay him to study more thereby breaking the chain. You see these kinds of process theories in many behavioral science literature. The treatment affected behavior, not because it directly changed behavior but because it changed a psychological, or internal state, and in turn that new psychological state contributed to the change in behavior. Different literatures vary on how much value they place on breaking open the black box and testing the intervening variables versus being content with just the stimulus-response association ("I pay my son to study and he gets better grades"). Under this viewpoint, I don't care why it happens, I think my money is well-spent if he gets better grades. As long at it works, I'll continue applying the intervention though there may be unintended consequences of this intervention such as potentially undermining his interest in the material because, after all, he is studying "for the money".

There is a philosophical debate that has lingered for decades. In the early days of social science there were many theories that made use of intervening variables attempting to explain the relation between two other variables. Often, those intervening variables were not directly observable. Analogous concepts have also appeared in the physical sciences, such as the original notion of gravity as an intervening variable between two observable variables even though the intervening variable is not directly observable. A competing view emerged in science that concepts should be based on observations otherwise they do not belong in science. In psychology this led to approaches such as behaviorism and logical positivism. The last few decades has seen an increase in attempting to measure such intervening variables and include them directly in the modeling approach. The hot modern version of intervening variables appears in deep learning models that make have impressive prediction accuracy but have multiple hidden layers (intervening variables between intervening variables between intervening variables etc) and the less sexy cousin PCA and factor analysis that model components

and factors that connect observables even though those variables are not directly observable.

Testing mediation is tricky. The modern method is to run an SEM model and test the product of two βs (the beta in the path from X to M times the beta in the path from M to Y). Testing products of two betas is not easy, and has led to specialized methods using bootstrap and Bayesian approaches.

Even if one can conduct a randomized experiment by manipulating X, it isn't clear that the randomized experimental approach can produce causal explanations in this setup because the relation between M and Y isn't manipulated when X is manipulated. If one manipulates both X and M, then that isn't directly testing mediation—one could test, for example, for main effects and interaction of X and M on Y. But that isn't testing the intervening variable proposition that X influences M (you tell me a secret) and in turn M influences Y (I then tell the secret to your advisor). So manipulation doesn't necessarily help with these kind of process models.

We should offer an entire course on the variants of mediation analysis, and there are complete textbooks just on mediation (a solid one is by David Mackinnon). Here, I just want to introduce the concept of mediation so you have some familiarity if you encounter it in your research literature. There are many details one needs to consider when running mediation-like tests, such as there are several models (ways of drawing arrows) that lead to different interpretations yet they are indistinguishable because they have the identical fit criteria. That is, different models imply the same covariance matrix, so data cannot distinguish different representations without additional assumptions or structure. An example is three variables X, M and Y were collected at the same time, then it is difficult to disentangle directions of arrows making it difficult to interpret the results along causal mechanisms without making assumptions about causal ordering. However, if X was collected at time 1, M was collected at time 2 and Y was collected at time 3, then Y could not have been a direct cause of M and consequently we can rule out that possible relation.

Mediation in R

Here is one approach to testing mediation in R using the lavaan package. I'm also using lavaan here as a way to gently introduce you to SEM modeling. We first define the structural model, which can be a set of several structural models. Here we have two structural submodels: one model for the mediator and a second model for the dependent variable. I'm using the letters a, b and c as labels for the betas so that I can then reuse the later to define new estimates like the product of a and b.[13] The analogous operation can be done in SPSS through the AMOS package but I prefer not to use AMOS because the syntax aspect is too cumbersome and the pull-

[13] Another way to perform mediation in R is through the mediation package, which has some addition features specific to mediation, including Bayesian methods, dealing with binary, count and nominal data, and computing various effect estimates.

down menu system to build a model is too clunky and can easily lead to errors in model specification. I'll show an example of AMOS syntax later in this notes and you'll see what I mean. An alternative way to test mediation in SPSS is through the PROCESS macros available through Hays website `https://processmacro.org/index.html` (also availalbe for R).

Here are the two structural models needed for the lavaan package. I wrote it across multiple lines so you can see the parts. The model is the entire set of expressions between the quotation marks.

```
library(lavaan)
model <- "
#structural model for the mediator
ME ~ a*T

#structural model for the dv
CP ~ b*ME + c*T

#extra parameters to compute for est, se(est), t and CI
indirect := a*b
direct := c
total := direct + indirect
"
```

The structural model is submitted to the sem() program in lavaan and summarized. The summary() output includes the estimates and tests of the indirect, direct and total effects. This R code chunk also includes a plot of the model with estimates and some additional summary using the bootstrap for confidence intervals.

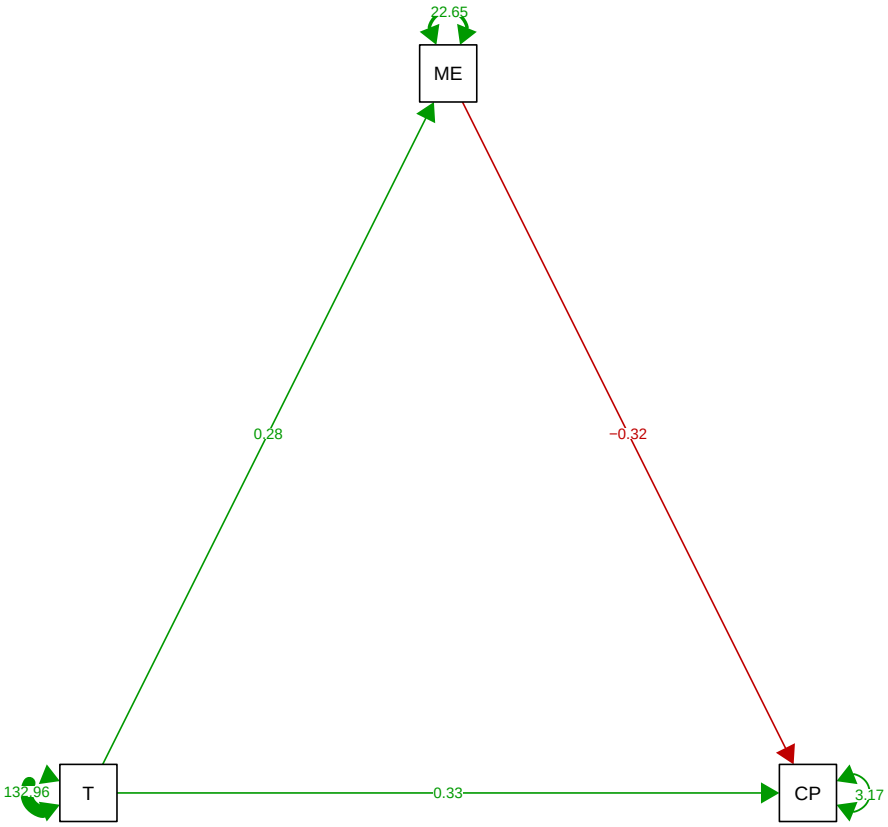
```
# estimate model, could include se='bootstrap'
fit <- sem(model, data = data, estimator = "ML")
summary(fit, fit.measures = T, rsq = T)
```

```
## lavaan 0.6.17 ended normally after 1 iteration
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                         5
##
##   Number of observations                            20
##
## Model Test User Model:
```

```
##
##   Test statistic                                0.000
##   Degrees of freedom                                0
##
## Model Test Baseline Model:
##
##   Test statistic                               36.603
##   Degrees of freedom                                3
##   P-value                                       0.000
##
## User Model versus Baseline Model:
##
##   Comparative Fit Index (CFI)                   1.000
##   Tucker-Lewis Index (TLI)                      1.000
##
## Loglikelihood and Information Criteria:
##
##   Loglikelihood user model (H0)               -99.487
##   Loglikelihood unrestricted model (H1)       -99.487
##
##   Akaike (AIC)                                208.975
##   Bayesian (BIC)                              213.953
##   Sample-size adjusted Bayesian (SABIC)       198.540
##
## Root Mean Square Error of Approximation:
##
##   RMSEA                                         0.000
##   90 Percent confidence interval - lower        0.000
##   90 Percent confidence interval - upper        0.000
##   P-value H_0: RMSEA <= 0.050                      NA
##   P-value H_0: RMSEA >= 0.080                      NA
##
## Standardized Root Mean Square Residual:
##
##   SRMR                                          0.000
##
## Parameter Estimates:
##
##   Standard errors                             Standard
##   Information                                 Expected
##   Information saturated (h1) model          Structured
##
## Regressions:
##                    Estimate  Std.Err  z-value  P(>|z|)
```

```
##   ME ~
##     T           (a)    0.284    0.092    3.075    0.002
##   CP ~
##     ME          (b)   -0.316    0.084   -3.783    0.000
##     T           (c)    0.335    0.042    7.991    0.000
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##    .ME               22.654    7.164    3.162    0.002
##    .CP                3.167    1.001    3.162    0.002
##
## R-Square:
##                    Estimate
##     ME                0.321
##     CP                0.764
##
## Defined Parameters:
##                    Estimate  Std.Err  z-value  P(>|z|)
##     indirect          -0.090    0.038   -2.386    0.017
##     direct             0.335    0.042    7.991    0.000
##     total              0.245    0.045    5.417    0.000
```

```
semPaths(fit, what = "par", fade = FALSE, layout = "spring")
```




```
# print bootstrap CIs for each parameter
parameterEstimates(fit, boot.ci.type = "bca.simple", level = 0.95,
    ci = TRUE, standardized = FALSE)
```

```
##        lhs op            rhs    label     est    se      z
## 1       ME  ~              T        a   0.284 0.092  3.075
## 2       CP  ~             ME        b  -0.316 0.084 -3.783
## 3       CP  ~              T        c   0.335 0.042  7.991
## 4       ME ~~             ME           22.654 7.164  3.162
## 5       CP ~~             CP            3.167 1.001  3.162
## 6        T ~~              T          132.962 0.000     NA
## 7 indirect :=          a*b indirect  -0.090 0.038 -2.386
## 8   direct :=            c   direct   0.335 0.042  7.991
```

```
## 9    total := direct+indirect    total   0.245 0.045  5.417
##   pvalue ci.lower ci.upper
## 1  0.002    0.103    0.465
## 2  0.000   -0.480   -0.152
## 3  0.000    0.253    0.417
## 4  0.002    8.613   36.695
## 5  0.002    1.204    5.129
## 6     NA  132.962  132.962
## 7  0.017   -0.163   -0.016
## 8  0.000    0.253    0.417
## 9  0.000    0.156    0.333
```

The indirect parameter estimate is defined as the product of a and b, the direct parameter estimate is the regression slope for the independent variable controlling for the mediator, and the total effect is the sum of the direct and indirect effects. This example is somewhat unusual in that the indirect effect is close to zero (slightly negative and significantly different from zero) but the two paths a and b are of different sign. The treatment T is associated with higher scores in ME and CP, even though the relation between ME and CP is negative. It is worth thinking about what this could mean, implications that can be derived and additional data that can test some of those predictions.

You can use covariance algebra to prove that the indirect effect is the product of a and b, and the total effect is the sum of direct and indirect (left as an exercise for those who are interested). The total effect is equivalent to the regression slope of CP (dv) on T (key predictor), as shown below. The idea of mediation is to decompose the total effect into two parts (direct and indirect) based on a model in order to assess the proportion of the effect that is directly related to the key independent variable versus the proportion of the effect that is indirectly related to the key independent variable through the mediator (the logic that the independent variable influences the mediator and then the mediator influences the dependent variable).

We can verify that the simple regression of T predicting CP estimates a slope that is equal to the total effect (direct plus indirect) in the mediation model. The parameter estimate for the slope out of the linear regression is equal to the total effect estimated in the SEM presented earlier.

```
coef(lm(CP ~ T, data = data))
```

```
## (Intercept)           T
##   77.014613    0.244872
```

mediation vs instrumental variables

It is instructive to compare mediation to the instrumental variable approach. These are different models even though the diagrams look similar. I'll write an SEM model following the instrumental variables logic where T is a predictor of ME, and we estimate a second regression with T as a predictor of CP. Recall the definition of the instrumental variables approach that is a ratio of two betas, or equivalently, a ratio of two mean differences when the instrument is a binary indicator of group membership. Dividing these two betas yields the estimate of the instrumental variable approach as per earlier in the lecture notes. There isn't a direct way of getting the SEM model to compute a slope using the predicted value, instead SEM wants to use a regression approach and compute the part correlation (i.e., controlling for the other variables in the regression). I implemented instrumental variables in SEM in a way that sidesteps this complication by computing the ratio of betas, following the definition of instrumental variables on page 13-40.

```
model2 <- "
#structural model for ME
ME ~ beta1*T

#structural model for CP
CP ~ beta2*T

#inst variable is b/a as per lecture notes
iv := beta2/beta1
"

# estimate model, could include se='bootstrap'
fit2 <- sem(model2, data = data, estimator = "ML")
summary(fit2, fit.measures = T, rsq = T)
```

```
## lavaan 0.6.17 ended normally after 16 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                         5
##
##   Number of observations                            20
##
## Model Test User Model:
##
##   Test statistic                                 0.000
##   Degrees of freedom                                 0
##
## Model Test Baseline Model:
##
```

```
##   Test statistic                                36.603
##   Degrees of freedom                                3
##   P-value                                       0.000
##
## User Model versus Baseline Model:
##
##   Comparative Fit Index (CFI)                   1.000
##   Tucker-Lewis Index (TLI)                      1.000
##
## Loglikelihood and Information Criteria:
##
##   Loglikelihood user model (H0)               -99.487
##   Loglikelihood unrestricted model (H1)       -99.487
##
##   Akaike (AIC)                                208.975
##   Bayesian (BIC)                              213.953
##   Sample-size adjusted Bayesian (SABIC)       198.540
##
## Root Mean Square Error of Approximation:
##
##   RMSEA                                         0.000
##   90 Percent confidence interval - lower        0.000
##   90 Percent confidence interval - upper        0.000
##   P-value H_0: RMSEA <= 0.050                      NA
##   P-value H_0: RMSEA >= 0.080                      NA
##
## Standardized Root Mean Square Residual:
##
##   SRMR                                          0.000
##
## Parameter Estimates:
##
##   Standard errors                             Standard
##   Information                                 Expected
##   Information saturated (h1) model          Structured
##
## Regressions:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   ME ~
##     T       (bet1)    0.284    0.092    3.075    0.002
##   CP ~
##     T       (bet2)    0.245    0.045    5.417    0.000
##
## Covariances:
```

```
##                    Estimate  Std.Err  z-value  P(>|z|)
##  .ME ~~
##    .CP               -7.165    2.953   -2.426    0.015
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##    .ME               22.654    7.164    3.162    0.002
##    .CP                5.433    1.718    3.162    0.002
##
## R-Square:
##                    Estimate
##     ME                0.321
##     CP                0.595
##
## Defined Parameters:
##                    Estimate  Std.Err  z-value  P(>|z|)
##     iv                0.863    0.402    2.145    0.032
```

Here is a different approach that also yields the instrumental variables estimate. This version does not directly create a ratio of betas, but instead by not including T as a predictor of CP bypasses the part correlation computation (i.e., the slope from ME to CP is computed with ME hat, the predicted value, rather than the residual of ME as in the case of mediation).

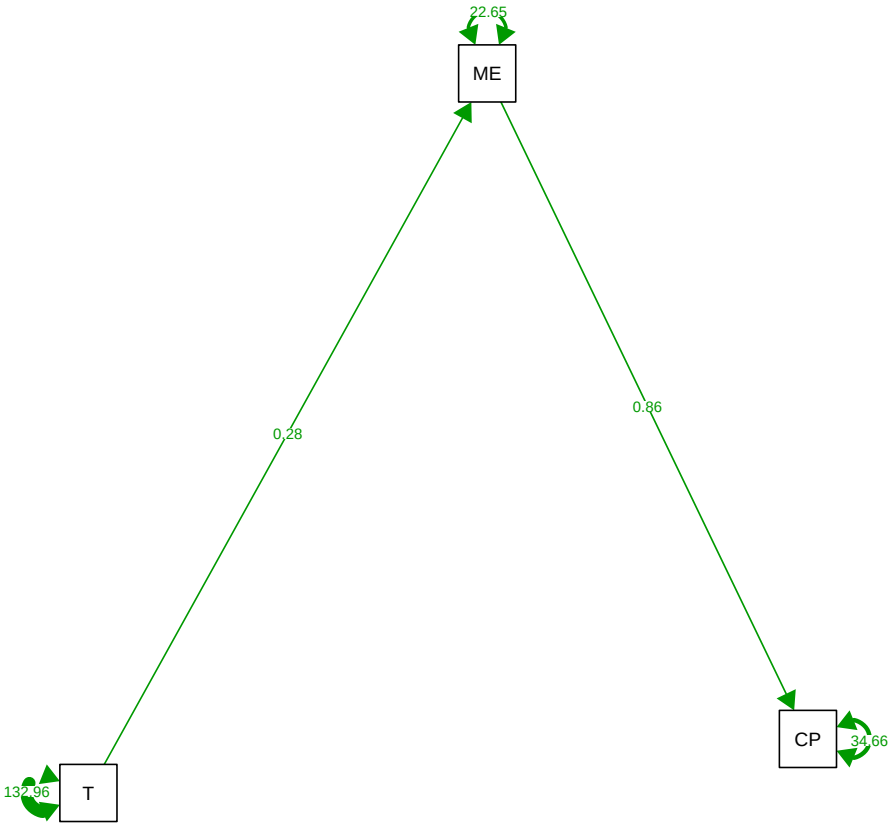
```
model3 <- "
ME ~ T
CP ~ ME

#estimate covariance between residual CP variance and residual ME variance
CP ~~ ME

"
fit3 <- sem(model3, data = data, estimator = "ML")
summary(fit3, fit.measures = T, rsq = T)
```

```
## lavaan 0.6.17 ended normally after 50 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                         5
##
##   Number of observations                            20
```

```
##
## Model Test User Model:
##
##   Test statistic                                0.000
##   Degrees of freedom                                0
##
## Model Test Baseline Model:
##
##   Test statistic                               36.603
##   Degrees of freedom                                3
##   P-value                                       0.000
##
## User Model versus Baseline Model:
##
##   Comparative Fit Index (CFI)                   1.000
##   Tucker-Lewis Index (TLI)                      1.000
##
## Loglikelihood and Information Criteria:
##
##   Loglikelihood user model (H0)               -99.487
##   Loglikelihood unrestricted model (H1)       -99.487
##
##   Akaike (AIC)                                208.975
##   Bayesian (BIC)                              213.953
##   Sample-size adjusted Bayesian (SABIC)       198.540
##
## Root Mean Square Error of Approximation:
##
##   RMSEA                                         0.000
##   90 Percent confidence interval - lower        0.000
##   90 Percent confidence interval - upper        0.000
##   P-value H_0: RMSEA <= 0.050                      NA
##   P-value H_0: RMSEA >= 0.080                      NA
##
## Standardized Root Mean Square Residual:
##
##   SRMR                                          0.000
##
## Parameter Estimates:
##
##   Standard errors                             Standard
##   Information                                 Expected
##   Information saturated (h1) model          Structured
##
```

```
## Regressions:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   ME ~
##     T                 0.284    0.092    3.075    0.002
##   CP ~
##     ME                0.863    0.402    2.145    0.032
##
## Covariances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##  .ME ~~
##    .CP              -26.712   12.570   -2.125    0.034
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##    .ME               22.654    7.164    3.162    0.002
##    .CP               34.665   24.127    1.437    0.151
##
## R-Square:
##                    Estimate
##     ME                0.321
##     CP               -1.586
```

```
semPaths(fit3, what = "par", fade = FALSE, layout = "spring")
```



The reason the standard error and z test are slightly different in the instrumental variables estimated using SEM compared to both the SPSS and R approaches presented earlier is that lavaan uses maximum likelihood (ML) rather than restricted maximum likelihood (REML) as compared to the usual instrumental variables setting. The implication is that SEM doesn't address degrees of freedom in the same way that regression approaches do, but as sample size increases these differences become very small. In the limit as N gets large, REML approaches ML.

FYI: An early paper by Smith (1982, JPSP, Beliefs, Attributions, and Evaluations: Nonhierarchical Models of Mediation in Social Cognition) combined both the logic of instrumental variables and the logic of mediation to be able to estimate models with reciprocal causality (A causes B and B causes A). He combined experimental manipulation in the context of an instrumental variable approach and the logic of

mediation of measured variables. This method hasn't received much attention but is an interesting way to use both instrumental and mediation methods simultaneously.

The IV literature has correctly addressed the issue of correlated residuals. However, related issues of measurement error have been pretty much ignored in the mediation literature (though mentioned in the original Baron & Kenney article). As we saw earlier in these lecture notes, measurement error in the context of a multiple regression can create serious problems, and in the setting of mediation we have multiple predictors.

15. Structural Equations Modeling

SEM is a general procedure that can do all ANOVA, regression, PCA, factor analysis, MANOVA, and canonical correlation. The framework allows for measurement error, so can handle the issues raised earlier in these lecture notes around multiple regression. It is possible to mix and match these techniques, such as perform a PCA with measurement error within a regression model to compare group differences.

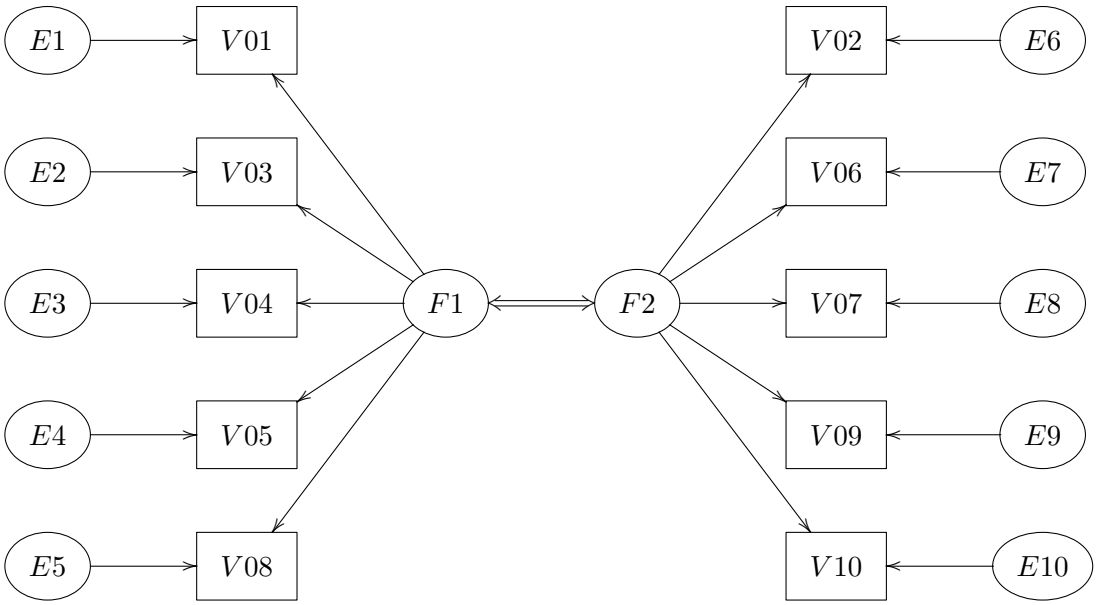
Words of Caution: Structural equations programs such as LISREL, EQS, CALIS, EZ-PATH, Amos, Mplus, lavaan and the like can be tricky. It is easy for an inexperienced user (and sometimes even an experienced user) to make a mistake and not be aware that he or she has made one. It is tempting to play with structural/measurement models (add a path here, remove a path over there, correlate an error, free a variance, etc). One little change can have a dramatic impact on the subtle features of the model (e.g., identification). You should develop a deep distrust of structural equations models that appear in the literature. I hope we can instill an appreciation for the elegance and necessity of this type of modeling as well as the motivation for the need to seek further training in how to use them. My goal in this brief introduction is to develop the necessary intuition and understand the published papers you read.

I already showed how to use this SEM approach to estimate mediation models. I'll now show an application of SEM with canonical correlation (LN#12) but extended to include measurement error so that one can simultaneously assess the reliability of each observed variable along with the factor structure and the correlation between the factors. So, this application of SEM extends a well-known technique in a new direction to account for the reliability of the observed data.

16. Factor Models in SEM

Structural equations modeling provides a way to estimate correlations between latent variables, or factors, in a manner that can automatically take into account measurement error (i.e., one does not need to perform an additional computation to correct

Figure 13-3: Graphical illustration of a structural equation model on the five positive and five negative items for the self-esteem data.



for attenuation because the model automatically does it). The analysis automatically provides factor loadings and reliabilities along with the relation between the factors.

Recall the self-esteem example from Lecture Notes #12 where I illustrated canonical correlation between two sets of variables. Figure 13-3 illustrates the generalization in graphical form of canonical correlation to allow for measurement error. The addition is that there are now 10 circles labelled E1 to E10 pointing to their respective observed variables. For example, this shows that observed variable V01 has two arrows point to it: the unobserved factor F1 and the unobserved measurement error E1 thus representing the regression equation

$$\text{V01} \quad = \quad \beta\text{F1} + \text{E1}$$

There are nine more such equations for each of the remaining nine observed variables V02 to V10. Each of these equations separately assesses the measurement error of each variable (the unobserved E) and the common component, the unobserved factor F. The parameters of the model include 10 βs, the variances of the two factors, and the variances of the unobserved errors (i.e., the variances of the 10 errors).

There are a few details to address such as the identification of parameters as we will soon see. But I'll present the overall structure of this model and its output. With the

big picture in hand using the self-esteem data, we will then examine the various parts of the output such as the goodness of fit tests using simpler examples.

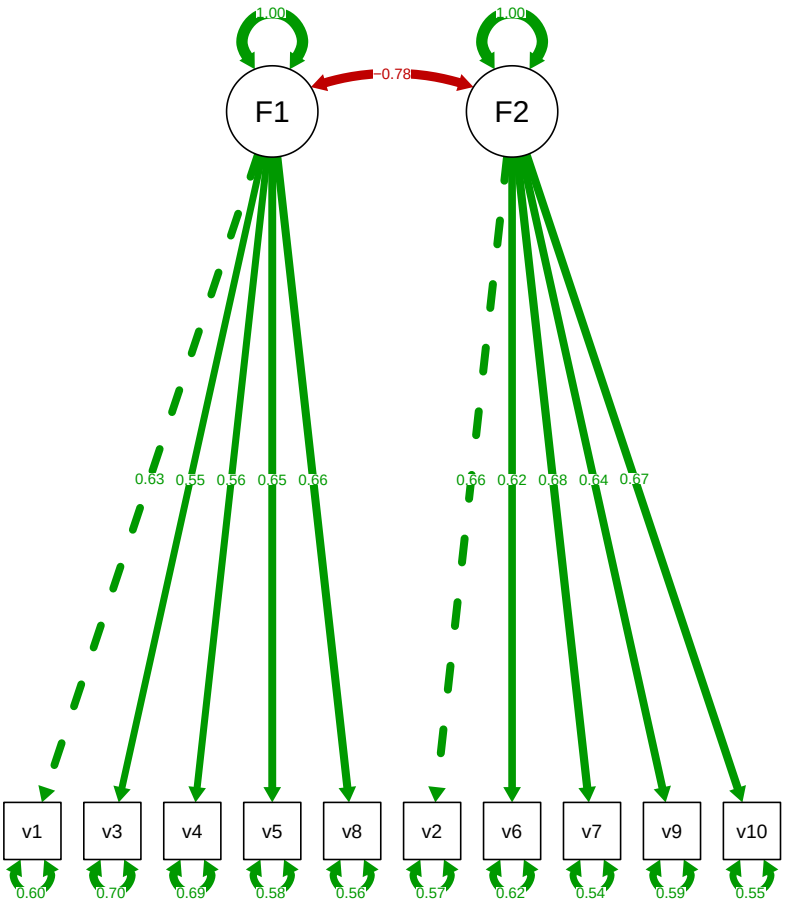
Here is an implantation of this more elaborate way to estimate the correlation between two factors in the context of measurement error. This is called *confirmatory factor analysis* (CFA)because we are examining the hypothesis that five observed variables load on the first factor and the other five observed variables load on the second factor. In other words, with 10 observed variables the first factor has five weights that are estimated on the five hypothesized observed variables with five weights set to 0 for the remaining observed variables. The second factor sets to 0 the weights on the observed variables associated with the first factor and estimates the weights associated with the remaining five observed variables. I'm using the lavaan package in R. The command is cfa() that stands for confirmatory factory analysis. One can also use Mplus, a commercial package, that has similar syntax as lavaan. I'll focus on lavaan in these lecture notes because it is open source. There are a few analyses that Mplus can do that lavaan cannot do yet, but all analyses that overlapped have been checked for consistency.

```
data <- read.table("../manova/selfest.dat", header = F)
# set matrix and assign variable names
data <- as.matrix(data)
colnames(data) <- c(paste("v", 1:10, sep = ""), "gender")

model <- "
F1 =~ v1 + v3 + v4 + v5 + v8
F2 =~ v2 + v6 + v7 + v9 + v10

#corr factor
F1 ~~ F2
"

library(lavaan)
# if you want only the standardized solution, add
# std.lv=TRUE
out.cfa <- cfa(model, data)

# saying standardized=T adds standardized estimates
summary(out.cfa, standardized = T, fit.measures = T, rsquare = T)


## lavaan 0.6.17 ended normally after 31 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                        21
```

```
##
##   Number of observations                           252
##
## Model Test User Model:
##
##   Test statistic                                118.465
##   Degrees of freedom                                34
##   P-value (Chi-square)                           0.000
##
## Model Test Baseline Model:
##
##   Test statistic                                785.166
##   Degrees of freedom                                45
##   P-value                                        0.000
##
## User Model versus Baseline Model:
##
##   Comparative Fit Index (CFI)                    0.886
##   Tucker-Lewis Index (TLI)                       0.849
##
## Loglikelihood and Information Criteria:
##
##   Loglikelihood user model (H0)              -2327.130
##   Loglikelihood unrestricted model (H1)      -2267.897
##
##   Akaike (AIC)                                4696.259
##   Bayesian (BIC)                              4770.377
##   Sample-size adjusted Bayesian (SABIC)       4703.804
##
## Root Mean Square Error of Approximation:
##
##   RMSEA                                          0.099
##   90 Percent confidence interval - lower         0.080
##   90 Percent confidence interval - upper         0.119
##   P-value H_0: RMSEA <= 0.050                    0.000
##   P-value H_0: RMSEA >= 0.080                    0.951
##
## Standardized Root Mean Square Residual:
##
##   SRMR                                           0.061
##
## Parameter Estimates:
##
##   Standard errors                             Standard
```

```
##   Information                                 Expected
##   Information saturated (h1) model          Structured
##
## Latent Variables:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   F1 =~
##     v1                1.000
##     v3                0.632    0.091    6.972    0.000
##     v4                0.864    0.122    7.103    0.000
##     v5                0.959    0.121    7.924    0.000
##     v8                0.720    0.089    8.059    0.000
##   F2 =~
##     v2                1.000
##     v6                0.975    0.120    8.103    0.000
##     v7                1.169    0.134    8.695    0.000
##     v9                0.967    0.116    8.313    0.000
##     v10               1.215    0.140    8.659    0.000
##    Std.lv  Std.all
##
##     0.584    0.631
##     0.369    0.546
##     0.504    0.559
##     0.560    0.647
##     0.420    0.663
##
##     0.377    0.657
##     0.367    0.619
##     0.440    0.677
##     0.364    0.639
##     0.457    0.674
##
## Covariances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   F1 ~~
##     F2               -0.172    0.028   -6.102    0.000
##    Std.lv  Std.all
##
##    -0.783   -0.783
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##    .v1               0.514    0.055    9.322    0.000
##    .v3               0.320    0.032   10.013    0.000
##    .v4               0.560    0.056    9.926    0.000
```

```
##    .v5                0.435    0.048    9.151    0.000
##    .v8                0.225    0.025    8.961    0.000
##    .v2                0.187    0.020    9.344    0.000
##    .v6                0.217    0.022    9.689    0.000
##    .v7                0.229    0.025    9.118    0.000
##    .v9                0.192    0.020    9.515    0.000
##    .v10               0.252    0.028    9.161    0.000
##     F1                0.341    0.068    4.996    0.000
##     F2                0.142    0.027    5.344    0.000
##    Std.lv  Std.all
##     0.514    0.602
##     0.320    0.702
##     0.560    0.688
##     0.435    0.581
##     0.225    0.560
##     0.187    0.569
##     0.217    0.617
##     0.229    0.541
##     0.192    0.592
##     0.252    0.546
##     1.000    1.000
##     1.000    1.000
##
## R-Square:
##                    Estimate
##     v1                0.398
##     v3                0.298
##     v4                0.312
##     v5                0.419
##     v8                0.440
##     v2                0.431
##     v6                0.383
##     v7                0.459
##     v9                0.408
##     v10               0.454


library(semPlot)
semPaths(out.cfa, what = "std", fade = FALSE)
```



It is helpful to compare the observed covariance matrix between these 10 variables to the model-implied covariance matrix, that is, using the model parameter estimates and the covariance algebra rules, we can compute the model-implied covariances. We can use the lavInspect() command with arguments sampstat for the observed covariance matrix and fitted for the model-implied covariance matrix. I also include the residual matrix (the difference between observed and model-implied matrices) to make it easier to spot discrepancies.

```
round(lavInspect(out.cfa, "sampstat")$cov, 2)
```

```
##        v1    v3    v4    v5    v8    v2    v6    v7    v9   v10
## v1   0.86
## v3   0.23  0.46
```

```
## v4   0.25  0.21  0.81
## v5   0.40  0.21  0.30  0.75
## v8   0.22  0.17  0.18  0.21  0.40
## v2  -0.22 -0.08 -0.20 -0.15 -0.13  0.33
## v6  -0.15 -0.09 -0.12 -0.12 -0.16  0.12  0.35
## v7  -0.21 -0.11 -0.25 -0.19 -0.17  0.22  0.17  0.42
## v9  -0.12 -0.09 -0.14 -0.12 -0.15  0.12  0.14  0.12  0.32
## v10 -0.17 -0.11 -0.16 -0.18 -0.16  0.14  0.18  0.17  0.24  0.46
```

```
round(lavInspect(out.cfa, "fitted")$cov, 2)
```

```
##        v1    v3    v4    v5    v8    v2    v6    v7    v9   v10
## v1   0.86
## v3   0.22  0.46
## v4   0.29  0.19  0.81
## v5   0.33  0.21  0.28  0.75
## v8   0.25  0.16  0.21  0.24  0.40
## v2  -0.17 -0.11 -0.15 -0.16 -0.12  0.33
## v6  -0.17 -0.11 -0.14 -0.16 -0.12  0.14  0.35
## v7  -0.20 -0.13 -0.17 -0.19 -0.14  0.17  0.16  0.42
## v9  -0.17 -0.11 -0.14 -0.16 -0.12  0.14  0.13  0.16  0.32
## v10 -0.21 -0.13 -0.18 -0.20 -0.15  0.17  0.17  0.20  0.17  0.46
```

```
# to make discrepancies easier to see look at residuals
round(lavInspect(out.cfa, "residuals")$cov, 2)
```

```
##        v1    v3    v4    v5    v8    v2    v6    v7    v9   v10
## v1   0.00
## v3   0.01  0.00
## v4  -0.04  0.02  0.00
## v5   0.07  0.01  0.02  0.00
## v8  -0.02  0.01 -0.03 -0.02  0.00
## v2  -0.04  0.02 -0.05  0.01 -0.01  0.00
## v6   0.02  0.02  0.02  0.04 -0.04 -0.02  0.00
## v7   0.00  0.02 -0.07  0.00 -0.02  0.05  0.00  0.00
## v9   0.05  0.02  0.00  0.04 -0.03 -0.02  0.00 -0.04  0.00
## v10  0.03  0.03  0.02  0.02 -0.01 -0.03  0.02 -0.04  0.07  0.00
```

This model perfectly reproduces the variances (diagonal terms) but some of the co-

variances are slightly off. This model has 21 parameters that were estimated (10 observed variances, 2 latent variances, 1 covariance between the latent variables, and 8 loadings). There still are 34 degrees of freedom left (degrees of freedom for these types of problems will be described later in these notes).

A note about working with SEM models, whether it be Mplus, lavaan or any other package. There has been relatively little investment in producing clear and concise output. In these lecture notes, I'll just print the output as it comes out of the box. In R there have been attempts to "prettify" the lavaan output but they don't work across all platforms, types of output (pdf, text, html) or types of lavaan models. Here is one example using the kable command in the knitr package to print the parameter estimates. This table is not pleasing to the eye but at least it doesn't take up so much space as the typical lavaan output. An analogous table could be developed for the goodness of fit measures.

```
knitr::kable(parameterEstimates(out.cfa), "latex", digits = 3)
```

lhs	op	rhs	est	se	z	pvalue	ci.lower	ci.upper
F1	=~	v1	1.000	0.000	NA	NA	1.000	1.000
F1	=~	v3	0.632	0.091	6.972	0	0.454	0.809
F1	=~	v4	0.864	0.122	7.103	0	0.626	1.103
F1	=~	v5	0.959	0.121	7.924	0	0.722	1.197
F1	=~	v8	0.720	0.089	8.059	0	0.545	0.896
F2	=~	v2	1.000	0.000	NA	NA	1.000	1.000
F2	=~	v6	0.975	0.120	8.103	0	0.739	1.211
F2	=~	v7	1.169	0.134	8.695	0	0.905	1.432
F2	=~	v9	0.967	0.116	8.313	0	0.739	1.195
F2	=~	v10	1.215	0.140	8.659	0	0.940	1.490
F1	~~	F2	-0.172	0.028	-6.102	0	-0.227	-0.117
v1	~~	v1	0.514	0.055	9.322	0	0.406	0.623
v3	~~	v3	0.320	0.032	10.013	0	0.258	0.383
v4	~~	v4	0.560	0.056	9.926	0	0.449	0.671
v5	~~	v5	0.435	0.048	9.151	0	0.342	0.528
v8	~~	v8	0.225	0.025	8.961	0	0.176	0.274
v2	~~	v2	0.187	0.020	9.344	0	0.148	0.226
v6	~~	v6	0.217	0.022	9.689	0	0.173	0.261
v7	~~	v7	0.229	0.025	9.118	0	0.179	0.278
v9	~~	v9	0.192	0.020	9.515	0	0.153	0.232
v10	~~	v10	0.252	0.028	9.161	0	0.198	0.306
F1	~~	F1	0.341	0.068	4.996	0	0.207	0.474
F2	~~	F2	0.142	0.027	5.344	0	0.090	0.194

It may be surprising to see a negative covariance and correlation between the two

factors given what we saw in LN12 under canonical correlation where the first cancor was .647. The sign change is due to the arbitrary signs of factor loadings. If you look at the cross-correlations (the correlations between variables in set 1 and set 2) you will see all the correlations are negative. The sign of the correlation is depending on the sign of the coefficients used to define the factors; if one factor is reverse coded relative to another factor, then the sign on the correlation will change. The R code in LN12 shows that one of the canonical correlation variables is defined with negative weights, which led to a positive canonical correlation. In the present example, both factors are coded with positive weights, so the covariance is negative.

There are many more fit measures that have been introduced in the literature. Here are all the ones printed by lavaan (check the help page for information on each of these fit measures).

```
fitmeasures(out.cfa)
```

```
##                  npar                  fmin
##                21.000                 0.235
##                 chisq                    df
##               118.465                34.000
##                pvalue        baseline.chisq
##                 0.000               785.166
##           baseline.df       baseline.pvalue
##                45.000                 0.000
##                   cfi                   tli
##                 0.886                 0.849
##                  nnfi                   rfi
##                 0.849                 0.800
##                   nfi                  pnfi
##                 0.849                 0.642
##                   ifi                   rni
##                 0.888                 0.886
##                  logl     unrestricted.logl
##             -2327.130             -2267.897
##                   aic                   bic
##              4696.259              4770.377
##                ntotal                  bic2
##              252.000              4703.804
##                 rmsea        rmsea.ci.lower
##                 0.099                 0.080
##        rmsea.ci.upper        rmsea.ci.level
##                 0.119                 0.900
##          rmsea.pvalue        rmsea.close.h0
##                 0.000                 0.050
```

```
## rmsea.notclose.pvalue     rmsea.notclose.h0
##                 0.951                 0.080
##                   rmr            rmr_nomean
##                 0.029                 0.029
##                  srmr          srmr_bentler
##                 0.061                 0.061
##   srmr_bentler_nomean                  crmr
##                 0.061                 0.067
##           crmr_nomean            srmr_mplus
##                 0.067                 0.061
##     srmr_mplus_nomean                 cn_05
##                 0.061               104.388
##                 cn_01                   gfi
##               120.254                 0.909
##                  agfi                  pgfi
##                 0.853                 0.562
##                   mfi                  ecvi
##                 0.846                 0.637
```

SEM and canonical correlation without "epsilon"

It turns out that the SEM implementation of canonical correlation in the traditional way without measurement error (i.e., without the "plus epsilon" I showed in LN12) is quite involved and not straightforward. Bagozzi, Fornell and Larcker (1981) seem to be the first to recognize this difficulty and used what is called a MIMIC representation to estimate the canonical correlation in the context of SEM. There are more recent papers (such as Fan, 1997, and Gu, Yung & Cheung, 2019) that show slightly different approaches but they are not necessarily simpler than Bagozzi et al. See those papers if you are interested. Here I give the basic logic using the approach in Fan, 1997, that makes use of both latent variables and composite variables. A latent variable has arrows going from the latent circle to the observed rectangles, whereas a composite variable has the arrows going in the opposite direction, from observed rectangles to latent circles. Some people use the terms *formative* and *reflective*, respectively. We have to run the model twice, switching with set of variables define the latent and composite variables, then use the weights from the composite definition to create the two weighted sums and correlated them. The models give a warning because this is not a traditional use of SEM.

```
#recall the two sets of variables to compute cancor
set1 <- data[,c(1,3,4,5,8)]
set2 <- data[,c(2,6,7,9,10)]

#first model has set 1 defining the composite
model.cancor1 <- '
F1 <~ v1 + v3 + v4 + v5 + v8
F1 =~ v2 + v6 + v7 + v9 + v10
```

```
#include all corrs across dvs
v2 + v6 + v7 +v9 ~~ v10
v2 + v6 + v7 ~~ v9
v2 + v6 ~~ v7
v2 ~~ v6

v1 + v3 + v4 +v5 ~~ v8
v1 + v3 + v4 ~~ v5
v1 + v3 ~~ v4
v1 ~~ v3
'

#second model has set2 defining the composite
model.cancor2 <- '
F1 =~ v1 + v3 + v4 + v5 + v8
F1 <~ v2 + v6 + v7 + v9 + v10

#include all corrs across dvs
v2 + v6 + v7 +v9 ~~ v10
v2 + v6 + v7 ~~ v9
v2 + v6 ~~ v7
v2 ~~ v6

v1 + v3 + v4 +v5 ~~ v8
v1 + v3 + v4 ~~ v5
v1 + v3 ~~ v4
v1 ~~ v3
'
#run both models
out.cancor1 <- sem(model.cancor1, data, estimator="ML",
                   std.lv=T, se="none")
out.cancor2 <- sem(model.cancor2, data, estimator="ML",
                   std.lv=T, se="none")
```

```
#extract weights from composite variables
weights.on.X <- coef(out.cancor1)[1:5]
weights.on.X


##      F1<~v1      F1<~v3      F1<~v4      F1<~v5      F1<~v8
## -0.22117978  0.02604075 -0.33872443 -0.09302677 -0.67169960


weights.on.Y <- coef(out.cancor2)[6:10]
```

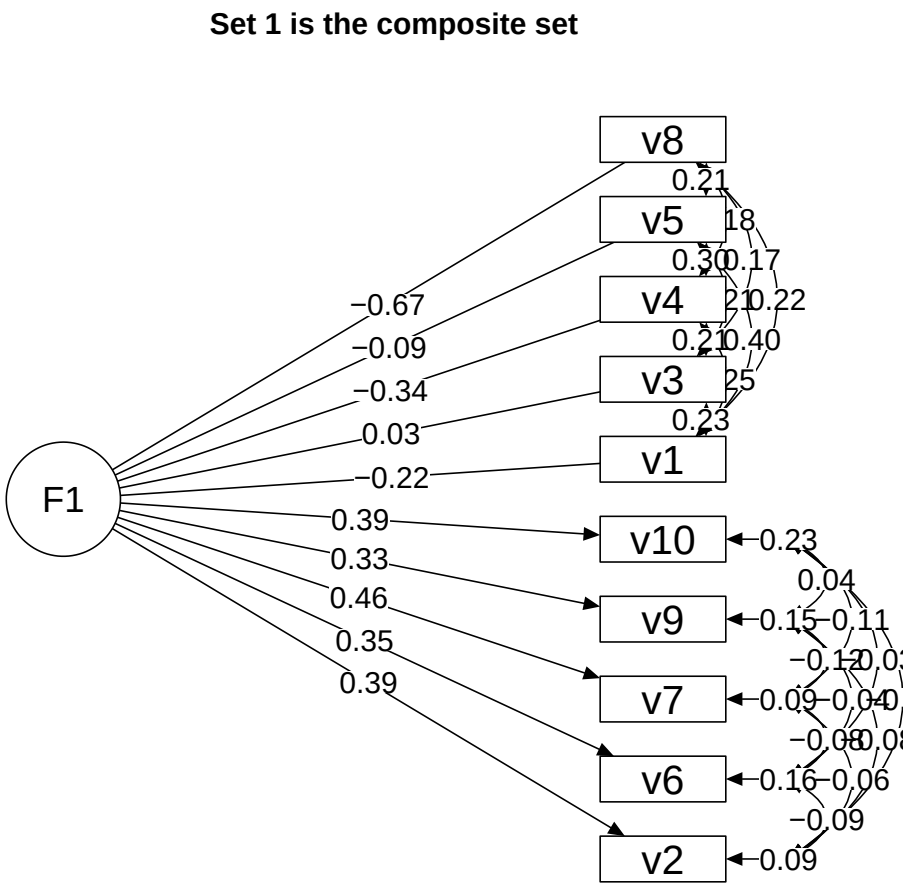
```
weights.on.Y
```

```
##     F1<~v2     F1<~v6     F1<~v7     F1<~v9    F1<~v10
## -0.4400753 -0.2867316 -0.4648510 -0.3381619 -0.1436342
```

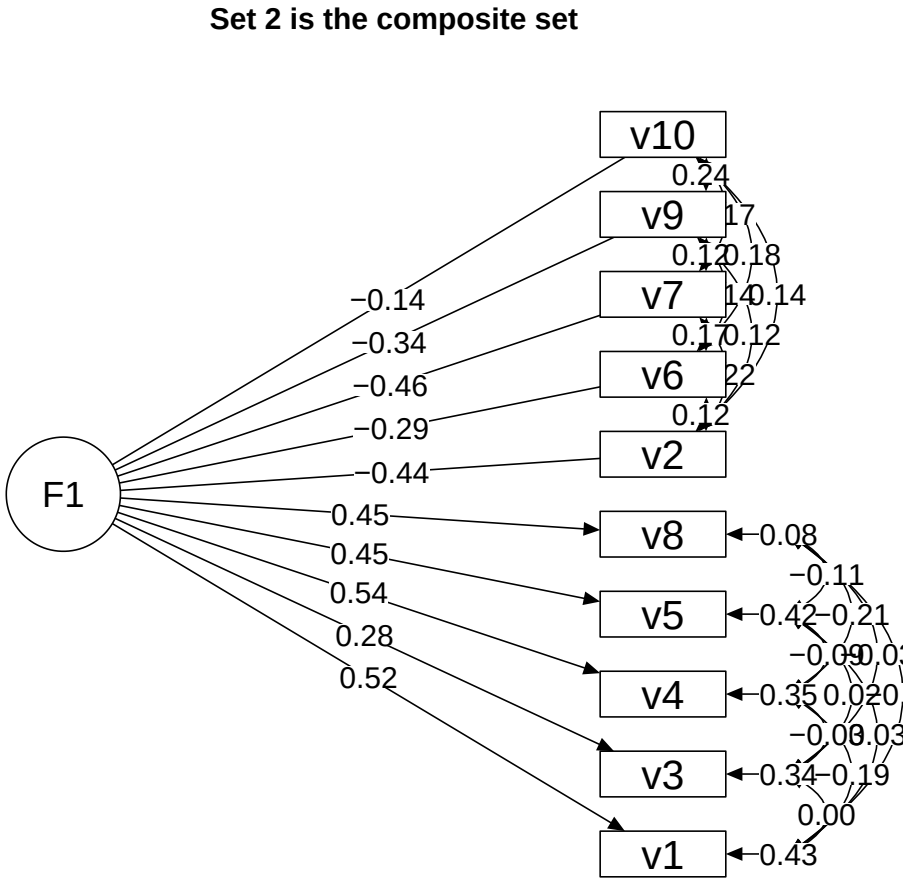
```
#create the two weighted sums and correlate
#same as the first canonical corr in LN12
cor(set1 %*% weights.on.X, set2 %*% weights.on.Y )
```

```
##             [,1]
## [1,] -0.6466087
```

```
#plot the two models
semPaths(out.cancor1, whatLabels = "est", layout = "tree2",
         rotation = 2, style = "lisrel", optimizeLatRes = TRUE,
         intercepts = FALSE, residuals = TRUE, curve = 1,
         curvature = 2,
         sizeLat = 10, nCharNodes = 8, sizeMan = 11, sizeMan2 = 4,
         edge.label.cex = 1.2, edge.color = "#000000")
title("Set 1 is the composite set")
```




```
semPaths(out.cancor2, whatLabels = "est", layout = "tree2",
         rotation = 2, style = "lisrel", optimizeLatRes = TRUE,
         intercepts = FALSE, residuals = TRUE, curve = 1,
         curvature = 2,
         sizeLat = 10, nCharNodes = 8, sizeMan = 11, sizeMan2 = 4,
         edge.label.cex = 1.2, edge.color = "#000000")
title("Set 2 is the composite set")
```



Now that we have the big picture of SEM in place, let's go through various pieces of the model and the output with simpler examples.

17. Assessing models

Recall that in regression models we assess the fit between the observed score y and the score predicted by the model, $\hat{y}$. A residual is the difference between the observed score and the predicted score:

$$\text{residual} \quad = \quad \text{observed score} - \text{predicted score} \tag{13-40}$$

In multiple regression the score predicted by the model is completely determined by the regression coefficients. One way to think about regression coefficients is that they

are the weights that minimize the sum of squared residuals (where the sum is taken over cases); that is, for each subject

$$\text{residual} \quad = \quad \text{observed score} - (\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots) \tag{13-41}$$

and the coefficients are selected so $\sum(\text{residual}^2)$ (sum of squared residuals over subjects) is minimized. The residual for a case is the estimated ϵ for that case.

This concept of difference between observed and predicted can be extended to covariance and correlation matrices.

$$\text{residual matrix} \quad = \quad \text{observed matrix} - \text{predicted matrix} \tag{13-42}$$

The difference is taken element by element. For example, assume the observed covariance matrix

	x	w	y
x	0.25	0.13	0.05
w	0.13	3.24	0.87
y	0.05	0.87	0.95

and the expected covariance matrix from some model

	x	w	y
x	0.25	0.30	0.15
w	0.30	3.56	1.46
y	0.15	1.46	1.37

The matrix of residuals is

	x	w	y
x	0.00	-0.17	-0.10
w	-0.17	-0.327	-0.59
y	-0.10	-0.59	-0.75

A matrix of residuals is not easy to interpret as is because the metric is not well-specified (e.g., is -.75 residual something to worry about?). So we need a metric that can be interpreted as goodness of fit of the model. Such an index is usually abbreviated GFI (goodness of fit index).

18. Goodness of fit (GFI)

The material in the next two sections is adapted from Bollen (1989) and Everitt (1984).

First, let's make a connection between GFI and the familiar concept of R^2. Recall that

$$R^2 = \frac{SSR}{SST} \quad (13\text{-}43)$$

$$= \frac{SST\text{-}SSE}{SST} \quad (13\text{-}44)$$

$$= 1 - \frac{SSE}{SST} \quad (13\text{-}45)$$

The last equation re-expresses R^2 as 1 minus the ratio of observed error over total error. The observed total error can be interpreted as a "baseline" to compare the observed error. Note that the baseline (the denominator) is based on observed scores not expected scores.

The sum of the squared elements in the residual matrix is a nice candidate for the analog to SSE. Square every element in the residual matrix, then sum all the squared elements. The analog to SST is just the sum of the squared elements in the observed covariance matrix (yes, squared variances). This leads to

$$GFI = 1 - \frac{\sum \text{squared terms from residual matrix}}{\sum \text{squared terms from observed cov matrix}} \quad (13\text{-}46)$$

How is this index interpreted? When the model is perfect, then the sum of squared residuals will be 0 and GFI = 1. When the model is "terrible" in the sense that the model predicts the same number for every subject, the sum of squared residuals will equal the sum of squared terms from the covariance matrix. Note that this "terrible" model could arise in cases where you know nothing and the best prediction for each subject on each variable is the mean of that variable. It is possible for this index to be negative.

NOTE: this particular GFI index is now outdated and has been replaced with better indices. The new indices are computationally intensive and are not as illuminating as the one presented here. See Bollen (1989) for newer treatments. Most GFI measures are based on the same underlying intuition. The reason I presented this form is because it illustrates the ideas in the simplest way, even though it is may not be the optimal way to define the concept.

19. Testing Fit

A simple Chi-square test for testing the fit of the model is

$$X^2 = \frac{(\mathrm{N}-1)}{2}\text{sum of squared terms from residual matrix} \qquad (13\text{-}47)$$

The computed X^2 can be compared to the tabled Chi-square value. The degrees of freedom are

$$\frac{k(k+1)}{2} - p \qquad (13\text{-}48)$$

where k is the total number of variables (i.e., number of rows or columns in the covariance matrix) and p is the number of parameters that are being estimated (e.g., the total number of regression coefficients that are used to compute the expected covariance matrix, not counting any parameter that is associated with error). The total number of parameters is simply the total number of slopes present in the model. Note that the intercepts don't enter here because adding or subtracting a constant has no effect on the variance or the covariance. A modification is needed to add intercepts and means to the mix of structural equations modeling, sometimes called mean-structure SEM.

There are peculiar things about this Chi-square test. First, the number of subjects does not enter into the computation of degrees of freedom. The reason is that what is being taken as data is the covariance matrix not the individual data points. The parameter k, the number of variables, captures the "size" of the covariance matrix which is a k x k matrix. Second, the null hypothesis here is the model you are trying to fit. So you want to accept the null hypothesis. A significant Chi-square means that the model you are testing does *not* fit the data, where by "fitting the data" we mean does the covariance matrix implied by the model match the covariance matrix of the data. Third, the sample size *does* enter into the computation of the actual Chi-square statistic, even though it did not enter into the GFI measure. So, with enough subjects (i.e., power) you will usually be able to reject the null hypothesis. There is an unusual tradeoff here: in order to be sure about accepting the null hypothesis you need lots of power but too much power might lead to erroneous rejection of a reasonable model.

NOTE: this particular Chi-square test is now outdated and has been replaced with better tests. The newer tests are computationally intensive and are not as illuminating as the one presented here. The newer tests get around some of the problems of this test. Also, some of the newer tests count parameters a little differently. We will discuss these newer tests later in the context of structural equations modeling.

20. Theoretical v. Empirical Models

If you know the true parameters, you can compute the expected covariance matrix and compare it to the observed covariance matrix using the GFI and the Chi-square test. However, typically one does not know the true parameters of the models. To

what matrix is the observed covariance matrix compared when the true model is not known?

The answer is easy. Assume a model, and run the regressions to estimate the relevant path coefficients. Now you have a model with numerical estimates. You can use the slopes estimated from these regression equations to create the model-implied covariance matrix. For example, suppose I was testing the model that w mediates the relation between x and y. Following the traditional approach to mediation by Baron & Kenny (1986) I would run two regressions: regress y on both w and x and regress x on w. Suppose the two regressions resulted in the following slopes and intercepts

$$y = 3.84 + .46w + .27x \quad (13\text{-}49)$$

$$w = -2.22 + 1.62x \quad (13\text{-}50)$$

With this information one can compute the model-implied covariance matrix and compare it to the observed covariance. That is, the regression output provides estimates that are used to compute the "expected" covariance matrix. This "expected" covariance matrix can be compared to the observed covariance matrix to see how well the model is doing in capturing the observed data. The GFI index and X^2 test can also be calculated. This is how the program works. Instead of known the regression betas, they are estimated in order to maximize GFI. That is, the program searches the optimal combination of betas that produce the expected covariance matrix that is closest to the observed covariance matrix.

In this way we can simultaneously assess the fit of several regression models. This is useful for more realistic testing where there may be several dependent variables and multiple hypotheses over these variables.

In this way, one use of SEM is to estimate several regression equations simultaneously (i.e., a system of regression equations).

21. SEM example

Let's compute an example with two factors on six observed variables with measurement error on each observed variable. I will use the six equations to set up a model that predicts the observed 6x6 covariance matrix. This model implied covariance matrix will be compared to the observed covariance matrix to estimate parameters (βs, factor variances, error variances, factor covariances). The six equations are listed here:

$$\text{x1A} = \beta 1\text{F1} + \text{E1} \quad (13\text{-}51)$$

$$\text{x1B} = \beta 2\text{F1} + \text{E2} \quad (13\text{-}52)$$

$$\text{x1C} = \beta 2\text{F1} + \text{E3} \quad (13\text{-}53)$$
$$\text{x2A} = \beta 3\text{F2} + \text{E4} \quad (13\text{-}54)$$
$$\text{x2B} = \beta 4\text{F2} + \text{E5} \quad (13\text{-}55)$$
$$\text{x2C} = \beta 4\text{F2} + \text{E6} \quad (13\text{-}56)$$

Here we have six observed variables, x1A to x2C. Each variable has its own error (denoted E1 to E6). There are two factors denoted F1 and F2. This can be interpreted as a set of six regression equations, where the goal is to estimate the six βs and the six error variances using two unknown factors. In this simple model we will set the variances of the unknown factors to 1 and force the two factors to be independent (i.e., have a covariance equal to 0). We will relax the second assumption (uncorrelated factors) later.

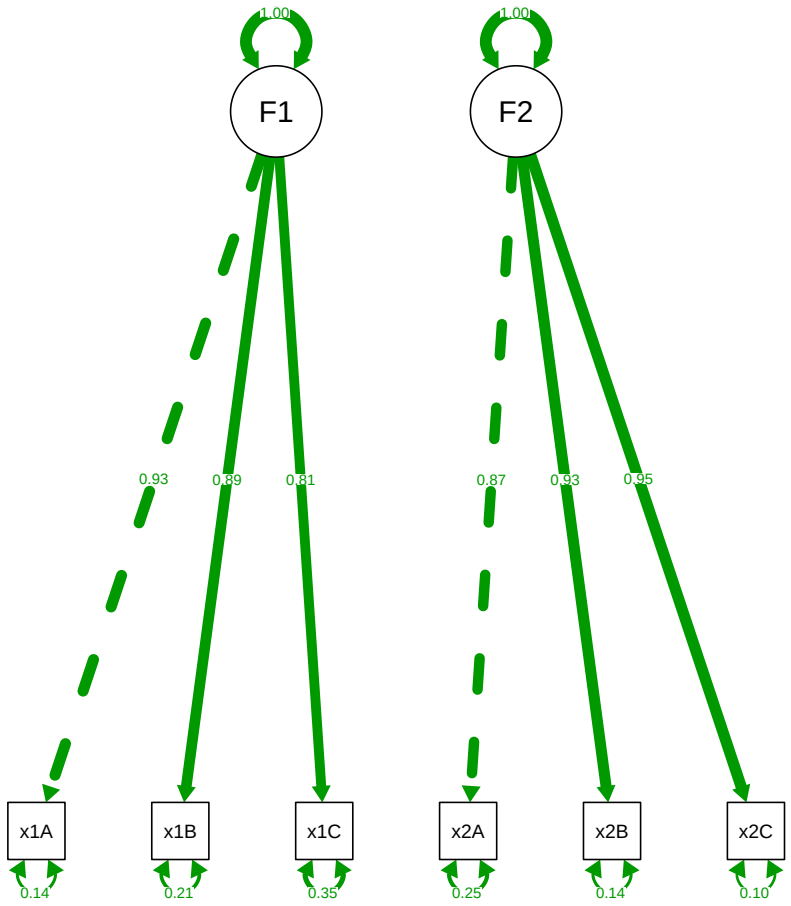
Here I'll use the R package lavaan. It is similar to the syntax used in MPlus making it easy to go back and forth (one can even run MPlus within R). I prefer to deviate from the traditional maximum likelihood estimate which uses biased variances/covariances (i.e., divide by N) rather than unbiased variances/covariances (i.e., divide by N - 1). Different programs can have different defaults so you should check the program you use so you understand the default settings. Mplus and R's lavaan both use the biased ML approach by default; below I specify likelihood="wishart" to yield the N - 1 version. The SPSS appendix shows AMOS output from SPSS.

```
library(lavaan)
data <- read.table("rg1.data")
# /users/gonzo/rich/Teach/multstat/unixfiles/lectnotes/sem/rg1.data')
colnames(data) <- c("id", "x1A", "x1B", "x1C", "x2A", "x2B",
    "x2C")

model1 <- "
#linear equations
F1 =~ x1A + x1B + x1C
F2 =~ x2A + x2B + x2C

#set variances and covariances
F1 ~~ F1
F2 ~~ F2
F1 ~~ 0*F2
"

out.sem <- sem(model1, data = data, likelihood = "wishart")
summary(out.sem, fit.measures = T)


## lavaan 0.6.17 ended normally after 24 iterations
```

```
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                        12
##
##   Number of observations                           120
##
## Model Test User Model:
##
##   Test statistic                                13.724
##   Degrees of freedom                                 9
##   P-value (Chi-square)                           0.132
##
## Model Test Baseline Model:
##
##   Test statistic                               576.533
##   Degrees of freedom                                15
##   P-value                                        0.000
##
## User Model versus Baseline Model:
##
##   Comparative Fit Index (CFI)                    0.992
##   Tucker-Lewis Index (TLI)                       0.986
##
## Loglikelihood and Information Criteria:
##
##   Loglikelihood user model (H0)                -880.980
##   Loglikelihood unrestricted model (H1)        -874.061
##
##   Akaike (AIC)                                1785.961
##   Bayesian (BIC)                              1819.411
##   Sample-size adjusted Bayesian (SABIC)       1781.472
##
## Root Mean Square Error of Approximation:
##
##   RMSEA                                          0.066
##   90 Percent confidence interval - lower         0.000
##   90 Percent confidence interval - upper         0.133
##   P-value H_0: RMSEA <= 0.050                    0.306
##   P-value H_0: RMSEA >= 0.080                    0.422
##
## Standardized Root Mean Square Residual:
##
##   SRMR                                           0.106
```

```
##
## Parameter Estimates:
##
##   Standard errors                             Standard
##   Information                                 Expected
##   Information saturated (h1) model          Structured
##
## Latent Variables:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   F1 =~
##     x1A               1.000
##     x1B               1.220    0.090   13.569    0.000
##     x1C               0.843    0.073   11.626    0.000
##   F2 =~
##     x2A               1.000
##     x2B               1.051    0.072   14.660    0.000
##     x2C               1.422    0.094   15.145    0.000
##
## Covariances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   F1 ~~
##     F2                0.000
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##     F1                1.089    0.170    6.389    0.000
##     F2                0.971    0.165    5.888    0.000
##    .x1A               0.176    0.057    3.121    0.002
##    .x1B               0.419    0.094    4.449    0.000
##    .x1C               0.416    0.065    6.377    0.000
##    .x2A               0.320    0.051    6.269    0.000
##    .x2B               0.179    0.040    4.444    0.000
##    .x2C               0.225    0.067    3.363    0.001
```

```
library(semPlot)
semPaths(out.sem, what = "std", fade = FALSE)
```




```
# residuals to compare obs to model-imp cov matrices
lavInspect(out.sem, "residuals")
```

```
## $cov
##       x1A   x1B   x1C   x2A   x2B   x2C
## x1A 0.000
## x1B 0.000 0.000
## x1C 0.000 0.000 0.000
## x2A 0.165 0.287 0.062 0.000
## x2B 0.255 0.406 0.082 0.000 0.000
## x2C 0.287 0.432 0.122 0.000 0.000 0.000
```

The model contains the six regression equations and specifications for the variances and covariances. For each observed variable we define a factor as a predictor and set up measurement error. The covariance between the two factors is fixed to 0 in this first run to yield two independent factors; a two-headed arrow does not appear between the two latent factors. The output reports goodness of fit (CFI and TLI) measures for this model that are related but not equivalent to the GFI presented earlier; the output also presents a Chi-square test that is a variant of the one presented earlier in these notes.

The output shows the estimates of the paths (one path in each factor is fixed to 1 for identification), the estimated factor variances and the six error variances. The residual covariance matrix shows that the covariances are accounted for perfectly within a factor but there are relatively large residuals in items across factors.

One may be interested in testing the correlation between the two factors. You run the same syntax except change the one line that fixes the correlation to 0 so that the program estimates the correlation.

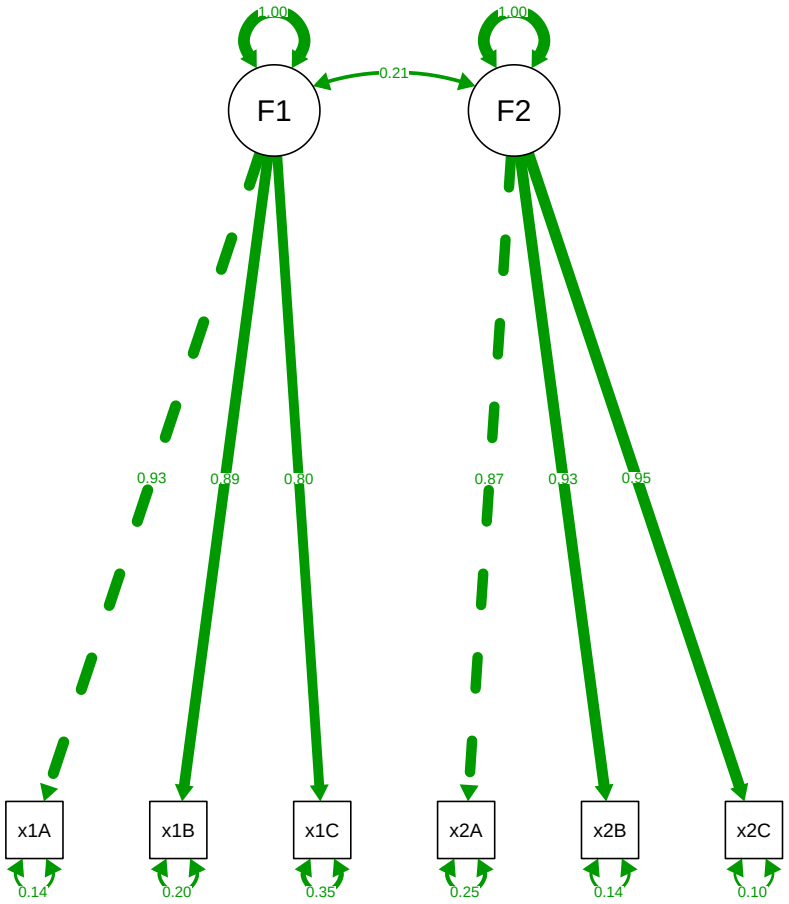
```
model2 <- "
#linear equations
F1 =~ x1A + x1B + x1C
F2 =~ x2A + x2B + x2C

#variances and covariances
F1 ~~ F1
F2 ~~ F2
F1 ~~ F2
"
out2.sem <- sem(model2, data = data, likelihood = "wishart")
summary(out2.sem, fit.measures = T, standardized = T)
```

```
## lavaan 0.6.17 ended normally after 25 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                        13
##
##   Number of observations                           120
##
## Model Test User Model:
##
##   Test statistic                                 9.255
##   Degrees of freedom                                 8
```

```
##   P-value (Chi-square)                                0.321
##
## Model Test Baseline Model:
##
##   Test statistic                                    576.533
##   Degrees of freedom                                     15
##   P-value                                             0.000
##
## User Model versus Baseline Model:
##
##   Comparative Fit Index (CFI)                         0.998
##   Tucker-Lewis Index (TLI)                            0.996
##
## Loglikelihood and Information Criteria:
##
##   Loglikelihood user model (H0)                    -878.727
##   Loglikelihood unrestricted model (H1)            -874.061
##
##   Akaike (AIC)                                     1783.454
##   Bayesian (BIC)                                   1819.691
##   Sample-size adjusted Bayesian (SABIC)            1778.591
##
## Root Mean Square Error of Approximation:
##
##   RMSEA                                               0.036
##   90 Percent confidence interval - lower              0.000
##   90 Percent confidence interval - upper              0.117
##   P-value H_0: RMSEA <= 0.050                         0.528
##   P-value H_0: RMSEA >= 0.080                         0.238
##
## Standardized Root Mean Square Residual:
##
##   SRMR                                                0.040
##
## Parameter Estimates:
##
##   Standard errors                                Standard
##   Information                                    Expected
##   Information saturated (h1) model          Structured
##
## Latent Variables:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   F1 =~
##     x1A               1.000
```

```
##     x1B               1.226    0.090   13.663    0.000
##     x1C               0.842    0.073   11.585    0.000
##   F2 =~
##     x2A               1.000
##     x2B               1.052    0.072   14.677    0.000
##     x2C               1.421    0.094   15.117    0.000
##    Std.lv  Std.all
##
##     1.042    0.926
##     1.278    0.894
##     0.877    0.804
##
##     0.985    0.867
##     1.037    0.927
##     1.400    0.946
##
## Covariances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   F1 ~~
##     F2                0.211    0.103    2.046    0.041
##    Std.lv  Std.all
##
##     0.206    0.206
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##     F1                1.086    0.170    6.381    0.000
##     F2                0.970    0.165    5.884    0.000
##    .x1A               0.179    0.056    3.195    0.001
##    .x1B               0.409    0.093    4.385    0.000
##    .x1C               0.420    0.066    6.413    0.000
##    .x2A               0.321    0.051    6.282    0.000
##    .x2B               0.176    0.040    4.403    0.000
##    .x2C               0.229    0.067    3.429    0.001
##    Std.lv  Std.all
##     1.000    1.000
##     1.000    1.000
##     0.179    0.142
##     0.409    0.200
##     0.420    0.353
##     0.321    0.249
##     0.176    0.141
##     0.229    0.105
```

```
semPaths(out2.sem, what = "std", fade = FALSE)
```




```
# residuals to compare obs to model-imp cov matrices
lavInspect(out2.sem, "residuals")
```

```
## $cov
##        x1A    x1B    x1C    x2A    x2B    x2C
## x1A  0.000
## x1B -0.002  0.000
## x1C  0.004  0.000  0.000
## x2A -0.046  0.028 -0.116  0.000
```

```
## x2B  0.032  0.133 -0.106 -0.001  0.000
## x2C -0.013  0.063 -0.131  0.002 -0.001  0.000
```

The covariance between the two factors is .206 and is statistically significant. The figure now draws a two-headed arrow between the two latent factors. The residuals between the observed and model-implied covariance matrices are now less extreme because the extra parameter for the correlated error between factors fits the data better.

It turns out that the test presented in most SEM programs for individual variables is incorrectly defined (I can supply a paper to those who are interested). The best way to test parameters such as the covariance or correlation between factors is to compare the Chi square tests across the two runs, the full and reduced model where the full model is when the parameter is free and the reduced model is when the parameter is set to the value of the null hypothesis (which is usually 0). Looking at these two outputs we have a Chisq of 9.25 with 8 degrees of freedom for the full model and a Chisq of 13.724 with 9 degrees of freedom for the reduced model. Obviously, the reduced model fits worse (a greater Chisq) because it has one fewer parameter to fit the data. There is one degree of freedom difference because the only change from the reduced to the full model is that we estimate one more parameter (the value of the covariance between the two factors). The difference between those two Chisq values is itself a Chisq distribution, so we can take 13.724 - 9.25 = 4.47, with one degree of freedom (9-8=1). You then look up the p-value for a Chisq of 4.47 with 1 d.f., which is about 0.035.

Under some conditions, the anova() command in R will correctly compare two lavaan models. You can see that the computation is equivalent to what I show by hand in the earlier paragraph.

```
anova(out.sem, out2.sem)
```

```
##
## Chi-Squared Difference Test
##
##          Df    AIC    BIC   Chisq Chisq diff   RMSEA
## out2.sem  8 1783.5 1819.7  9.2547
## out.sem   9 1786.0 1819.4 13.7240     4.4693 0.17003
##          Df diff Pr(>Chisq)
## out2.sem
## out.sem        1    0.03451 *
## ---
## Signif. codes:
```

```
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

22. What to Report from an SEM Run?

    What do you do after you run all these models? You would like to decide which model "fits" the data better so you need to supply the necessary information. You can organize the output in a way that facilitates comparison. For example, here is one suggestion. You could also list the residual matrix of each model and eye-ball to see where each of them fails. Once you decide on a best model, then you present the relevant parameter estimates and interpret the parameters of the best fitting model.

Model	# parameters	GFI	CFI	RMSEA	Chi-square	df
1						
2						
3						
4						

    Other measures that have risen in popularity are AIC and BIC.

    Keep in mind that there is a trade-off between number of parameters and how well a model fits. Are there any models that stick out as not being good in comparison to the rest? If so, those can be dropped unless you are trying to keep the number of parameters down.

    Recall that it is possible to compare two models by taking the difference between the two Chi-square values and subtracting the degrees of freedom from both models, always taking the difference in the order "reduced minus full". This works when one model is nested within the other (i.e., the two models are identical except that one or more paths in the full model are forced to zero in the reduced model).

    Whenever you have a standard error for a parameter, you can create a confidence interval around that parameter by using the usual "plus or minus 1.96 times the standard error" rule. But as I hinted above, the tests as defined in most SEM programs give poor approximations to the standard error of parameters. It is best to test parameters using the "reduced minus full" approach I showed above; that is, set the parameter to the value of the null hypothesis (reduced) and rerun the model with the parameter as a free parameter (full model). See Gonzalez & Griffin (2001) for an explanation for why this approach should be used over the other possible approaches.

23. SEM and Linear Mixed Models (aka Hierarchical Linear Models)

There is a simple connection between SEM and linear mixed models (also called multilevel models, or hierarchical linear models). We used these models in LN5 for repeated-measures designs and also to address fixed and random factors in ANOVA.

In terms of the standard SEM path diagram, we can think of the observed "squares" as nested within the unobserved "circles." Think back to ANOVA designs when we talked about random effects and nested effects. We discussed the repeated measures design as having a subjects factor that is treated as a random effect and repeated observations being connected to a given subject. For example, if there are three observations per subject we would write an ANOVA design as

$$Y_{ij} \quad = \quad \mu + \alpha_i + \pi_j + \epsilon_{ij} \tag{13-57}$$

with π a random effect representing subjects; it is random because we are generalizing to the population of subjects. The three time points are represented through the three αs. This model also gives us an interpretation that each subject has their own unique intercept.

We saw this same design in regression where we treated subjects as a factor and created N-1 dummy codes to properly account for the subject variance.

We can represent this as a structural equation model with latent variable representing the intercept term and slope terms.

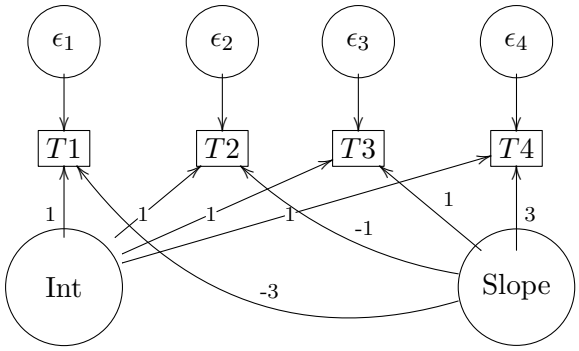
In LN5 we represented this as a hierarchical linear model. In SPSS we used the MIXED command and in R we used lme (though the appendix to LN5 shows the complication within R in getting the right error term and degrees of freedom). The general syntax of the MIXED command looks like this

```
MIXED DV with X1 X2 etc XK
 /print = solution
 /method=ml
 /fixed = X1 X2 etc XK
 /random = intercept X1 | subject(ID) covtype(UN).
```

This syntax has a dependent variable DV where the intercept and the slope for X1 are random and nested within the variable code ID with covtype(UN) meaning unstructured, or free, variance and covariance between intercept and random slope. The fixed part has a common slope for each subject. If the DV is a long column of data with all the time 1 data, then all the time 2 data, etc., and the Xs contain time contrasts (like the linear contrast -3 -1 1 3 for, say, four times), then we can extend the simple repeated measures ANOVA by allowing each subject to have a unique slope

and unique intercept. In this case, the simple $Y_{ij} = \mu + \alpha_i + \pi_j + \epsilon_{ij}$ model adds one more term to allow for a random slope per subject as well as the random intercept. We can estimate the variability of, say, the slope. In more complex models we can predict the variability or use the variability to predict other data. This allows us to model the heterogeneity in the data as we would model any other statistical process.

Now that the reminders are in place I switch to the SEM representation. The latent growth model can be represented in a standard SEM diagram like this:



with the unit contrast (1, 1, 1, 1) defining the latent variable called intercept and the contrast (-3,-1,1,3) defining the latent variable called slope. This example has four observed times on the same variable, labeled T1 to T4. By setting the factor paths to specific contrast values, the latent variables are defined just like contrast values in ANOVA. Each subject is assigned a contrast score (we called them "I hat" last semester) separately for intercept and for slope. Thus, each subject has his/her own slope and intercept. This SEM representation will accomplish the same thing as the MIXED model above. There are subtle differences though. The traditional latent growth curve model sets all error variances across time to be equal, but the SEM framework can estimate a more general model with separate error variances over time and more complicated error structures. Though, it is possible to set the error variances to be equal in SEM (or in the mixed model formulation to generalize the error variances to be unequal).

The lavaan syntax for a latent growth curve model is listed below. In this example we have a random intercept and a random slope (meaning each subjects is assigned their own intercept and slope). These are treated in lavaan by defining two latent factors, denoted i and s for intercept and slope, with the four observed times as the indicators and the contrast weights as fixed loadings. The intercept is defined by the unit contrast and the slope is defined by the linear contrast (-3,-1,1,3). Further, the example includes two predictors of those random intercepts and slopes (x1 and x2), something that is not easy to implement in traditional regression-based approaches

and highlights the benefits of using an SEM approach that can be so flexible. We use the special growth() command in lavaan to estimate the growth curve model, which has some additional constraints imposed relative to a completely general SEM; the growth() command also sets up the mean-structure version of SEM because recall that earlier in these lecture notes I pointed out that means and intercepts need to be treated in a special way. There is an analogous package in R called blavaan for computing Bayesian versions of SEM models. The MPLUS program, available through UM's site license program, can also run general SEMs and their Bayesian analogues.

```
# define growth curve model
model.syntax <- "
  # intercept and slope with fixed coefficients
    i =~ 1*t1 + 1*t2 + 1*t3 + 1*t4
    s =~ -3*t1 + -1*t2 + 1*t3 + 3*t4

  # regressions
    i ~ x1 + x2
    s ~ x1 + x2
"

# fit model and print summary
fit1 <- growth(model.syntax, data = Demo.growth)
summary(fit1, fit.measures = TRUE)
```

```
## lavaan 0.6.17 ended normally after 38 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                        13
##
##   Number of observations                           400
##
## Model Test User Model:
##
##   Test statistic                                10.873
##   Degrees of freedom                                 9
##   P-value (Chi-square)                           0.285
##
## Model Test Baseline Model:
##
##   Test statistic                              2209.146
##   Degrees of freedom                                14
##   P-value                                        0.000
##
```

```
## User Model versus Baseline Model:
##
##   Comparative Fit Index (CFI)                    0.999
##   Tucker-Lewis Index (TLI)                       0.999
##
## Loglikelihood and Information Criteria:
##
##   Loglikelihood user model (H0)              -2541.899
##   Loglikelihood unrestricted model (H1)      -2536.462
##
##   Akaike (AIC)                                5109.797
##   Bayesian (BIC)                              5161.686
##   Sample-size adjusted Bayesian (SABIC)       5120.436
##
## Root Mean Square Error of Approximation:
##
##   RMSEA                                          0.023
##   90 Percent confidence interval - lower         0.000
##   90 Percent confidence interval - upper         0.063
##   P-value H_0: RMSEA <= 0.050                    0.838
##   P-value H_0: RMSEA >= 0.080                    0.006
##
## Standardized Root Mean Square Residual:
##
##   SRMR                                           0.011
##
## Parameter Estimates:
##
##   Standard errors                             Standard
##   Information                                 Expected
##   Information saturated (h1) model          Structured
##
## Latent Variables:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   i =~
##     t1                1.000
##     t2                1.000
##     t3                1.000
##     t4                1.000
##   s =~
##     t1               -3.000
##     t2               -1.000
##     t3                1.000
##     t4                3.000
```

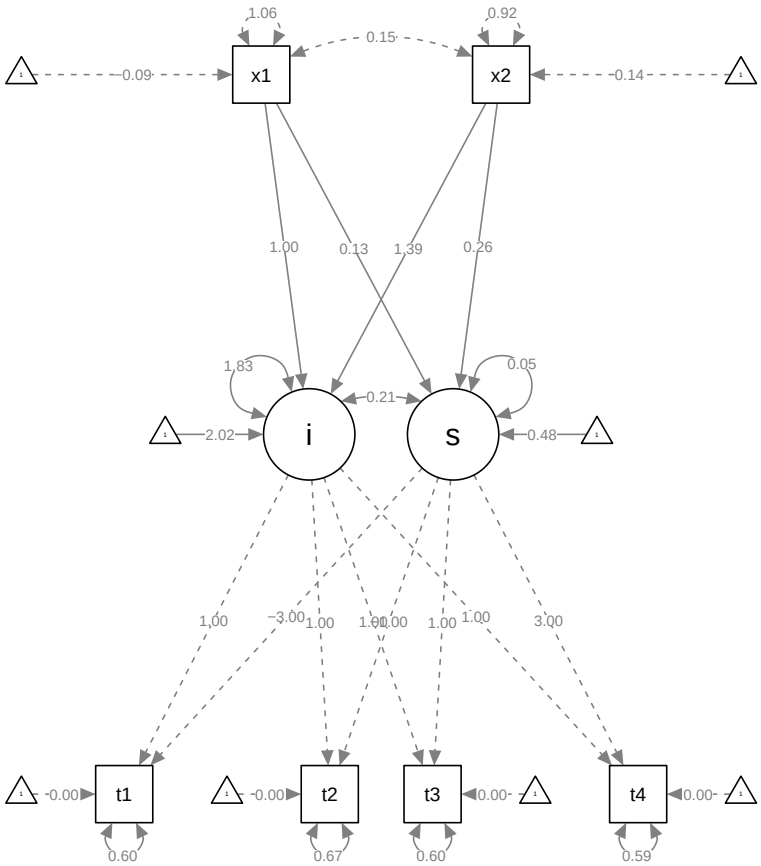
```
##
## Regressions:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   i ~
##     x1                1.003    0.069   14.447    0.000
##     x2                1.389    0.074   18.711    0.000
##   s ~
##     x1                0.131    0.014    9.157    0.000
##     x2                0.259    0.015   16.868    0.000
##
## Covariances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##  .i ~~
##    .s                 0.208    0.023    8.884    0.000
##
## Intercepts:
##                    Estimate  Std.Err  z-value  P(>|z|)
##    .i                 2.024    0.072   28.247    0.000
##    .s                 0.479    0.015   32.382    0.000
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##    .t1                0.597    0.085    7.067    0.000
##    .t2                0.671    0.060   11.088    0.000
##    .t3                0.599    0.065    9.254    0.000
##    .t4                0.593    0.109    5.437    0.000
##    .i                 1.833    0.141   13.020    0.000
##    .s                 0.055    0.007    7.975    0.000
```

```
# to get printout of random effects
output.ranef <- lavPredict(fit1, type = "lv")
# print first 10 subjects
head(output.ranef, n = 10)
```

```
##                i           s
##  [1,]  2.1034953  0.31512702
##  [2,] -4.7209684 -0.66187392
##  [3,]  1.0962164  0.44597157
##  [4,]  3.3266591  0.70577223
##  [5,] -0.4733445 -0.01088167
##  [6,] -0.5401452  0.26165794
##  [7,]  2.5506954  0.46428072
```

```
##  [8,] -1.7850512 -0.13194787
##  [9,]  2.4476691  0.49489328
## [10,]  1.7549396  0.50172964
```

```
semPaths(fit1, what = "path", whatLabels = "par", fade = FALSE,
    style = "ram", layout = "tree", intAtSide = TRUE)
```



The resulting graph looks complicated but that's because we ran a complicated model. Not only did we have a random intercept and random slope on the four time points t1-t4, but those random effects were also predicted by two other variables x1 and x2. Neither ANOVA nor regression can easily run this model where temporal latent

factors are both estimated from a repeated measures on t1-t4 and predicted by two other variables x1 and x2. This style graphic to depict the structural equations model was first proposed by J. McCardle. You already know about circles and squares and single-headed vs double-headed arrows. This model also includes intercepts that allow means to also be estimated. These intercepts are depicted by the triangles; in this model each observed and latent variable has a mean.

The number of "data points" for this model includes both the observed covariance matrix, which is a 6x6 matrix so has 21 variances and covariances, and six observed means for a total of 27 (21 = 6).

The total number of parameters estimated in the output are 4 regression slopes from the two X predictors to the random intercept and random slopes, 2 variances for the random intercept and random slope, 1 covariance between the two random effects, 2 means for the two random effects, and 4 variances for the four T observed variables. That is a total of 13 parameters as indicated in the output (4 + 2 + 1 + 2 + 4).

In addition, the default in SEM models is that exogenous intercepts, variances and covariances are automatically estimated but not printed in the output. In this model the two X predictor variables are exogenous because they do not have any single-headed arrows pointing to them other than the intercept. So that adds two means, two variances and one covariance for a total of 5 additional parameters.

Let's work out the degrees of freedom. There are 27 data points, 13 parameters explicitly called out in the output and 5 additional parameters implicitly estimated. That yields 9 degrees of freedom as printed in the output (27 - 13 - 5 = 9).

It may seem strange that SEM programs routinely don't print the parameters of the exogenous variables but still include them in the computation of the degrees of freedom. I agree. If you want those estimates included the output, you need to make them explicit in the model statement. I'll show that here. Note that there are now 18 parameters that are estimated in the model (the original 13 plus the additional 5 that I'm calling out the be included in the model output), but the model still reports 9 degrees of freedom and all the goodness of fit measures are identical to before. But there are minor changes to the baseline "independence" model, likelihood and the information criterion metrics by asking SEM to print out these estimates because of how SEM defines the baseline model. With some extra work this can be fixed in the printout so both representations use the same baseline model.

```
# define growth curve model
model.syntax <- "
  # intercept and slope with fixed coefficients
    i =~ 1*t1 + 1*t2 + 1*t3 + 1*t4
```

```
    s =~ -3*t1 + -1*t2 + 1*t3 + 3*t4

  # regressions
    i ~ x1 + x2
    s ~ x1 + x2

  # extra lines to print parameters associated with exogenous variables
  x1 ~~ x1
  x2 ~~ x2
  x1 ~~ x2
  x1 ~ 1
  x2 ~ 1
"

# fit model and print summary
fit2 <- growth(model.syntax, data = Demo.growth)
summary(fit2, fit.measures = TRUE)
```

```
##
##   Loglikelihood user model (H0)              -3667.158
##   Loglikelihood unrestricted model (H1)      -3661.722
##
##   Akaike (AIC)                                7370.317
##   Bayesian (BIC)                              7442.163
##   Sample-size adjusted Bayesian (SABIC)       7385.048
##
## Root Mean Square Error of Approximation:
##
##   RMSEA                                          0.023
##   90 Percent confidence interval - lower         0.000
##   90 Percent confidence interval - upper         0.063
##   P-value H_0: RMSEA <= 0.050                    0.838
##   P-value H_0: RMSEA >= 0.080                    0.006
##
## Standardized Root Mean Square Residual:
##
##   SRMR                                           0.011
##
## Parameter Estimates:
##
##   Standard errors                             Standard
##   Information                                 Expected
##   Information saturated (h1) model          Structured
##
## Latent Variables:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   i =~
##     t1                1.000
##     t2                1.000
##     t3                1.000
##     t4                1.000
##   s =~
##     t1               -3.000
##     t2               -1.000
##     t3                1.000
##     t4                3.000
##
## Regressions:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   i ~
##     x1                1.003    0.069   14.447    0.000
##     x2                1.389    0.074   18.711    0.000
```

```
##   s ~
##     x1                0.131    0.014    9.157    0.000
##     x2                0.259    0.015   16.868    0.000
##
## Covariances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   x1 ~~
##     x2                0.153    0.050    3.068    0.002
##  .i ~~
##    .s                 0.208    0.023    8.884    0.000
##
## Intercepts:
##                    Estimate  Std.Err  z-value  P(>|z|)
##     x1               -0.092    0.051   -1.793    0.073
##     x2                0.138    0.048    2.878    0.004
##    .i                 2.024    0.072   28.247    0.000
##    .s                 0.479    0.015   32.382    0.000
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##     x1                1.056    0.075   14.142    0.000
##     x2                0.924    0.065   14.142    0.000
##    .t1                0.597    0.085    7.067    0.000
##    .t2                0.671    0.060   11.088    0.000
##    .t3                0.599    0.065    9.254    0.000
##    .t4                0.593    0.109    5.437    0.000
##    .i                 1.833    0.141   13.020    0.000
##    .s                 0.055    0.007    7.975    0.000
```

With four time points it is possible to also have a quadratic and a cubic contrast (recall the number of contrasts is constrained by number of time points minus 1). But, one can't necessarily make all of those terms random effects because in typical repeated measures does not have sufficient data to estimate all the required variances. So, while the fixed effect terms can be estimated for linear, quadratic and cubic contrasts, it is likely that only the linear (and depending on the data, possibly the quadratic) can be treated as a random effect term. If one requests random effects on all terms, one can easily run into the problem that there are more terms to estimate than there data points. We saw this idea pop up at various stages in earlier Lecture Notes. There have been papers suggesting that all parameters be treated as random effects, but those papers are easily to dismiss because often those models cannot be estimated. If you try to estimate you will see weird output like NAs in the output (recall the examples I did in LN4 if you try to estimate an interaction when you only have one observation per cell).

All of these formulations—repeated measures, regression, SEM, multilevel model—will yield the identical results if they are each run in a particular way and there are no missing data. But the reason for the more complicated models is that they provide more flexibility, such as each subject has his or her own slope or better ways of handling missing data such as full-information maximum likelihood (FIML) and imputation. All specialized topics that would be part of a third semester course following 613/614.

As I mentioned earlier, there is a Bayesian version of lavaan called blavaan. In a Bayesian framework one needs to do careful diagnostics to spot problems of over-specifying the number of parameters in a model (such as the number of random effects, factors, terms properly identified, etc). The Bayesian programs will produce output even when the problem technically cannot be solved; hints that there are problems will appear in the diagnostics and other parts of the model, you just have to bother to look and, much like assumption testing, most people typically don't do the extra work.

There are more general formulations of these latent growth curve models that allow one to estimate subgroups of subjects that exhibit different values on the parameters of their trajectories. You can think of this as a way to cluster subjects based on their commonality on the parameter values (e.g., slope and intercept). This idea is similar to k-means and other clustering methods we reviewed in Lecture Notes 10. These more general models that allow subject clustering in the context of SEM are called "mixture models." The mixture model idea can be applied to any type of SEM, so we can have, for example, mixture models in mediation to assess whether there are subgroups of subject that exhibit different patterns of mediation, mixture models of factor analysis to assess whether subgroups of subjects exhibit different factor structures, and, as relevant to this subsection, mixture models of growth curve models to assess whether subjects cluster on different properties of growth (such as different intercepts and slopes). This requires using a specialized program Mplus as neither SPSS nor R can handle all the variations of mixture models. If I taught a third semester, I would include these mixture models along with some machine learning models, some more Bayesian analyses and some examples of "process models" (such as computational models and agent-based models).

24. Item Response Theory (IRT)

Earlier in these lecture notes I presented reliability on binary data (e.g., there judges each rate 8 subjects). I glossed over the fact that in those examples the data were binary, yet the key measures of reliability, like Cronbach's alpha, just plugged along as though the data were continuous and normally distributed.

There are newer extensions of the concept of reliability that are collectively known as Item Response Theory (IRT). These models can be fit through SEM as well so I'll

discuss them briefly.

[Need to add this subsection]

25. Coda

We end the course with the most difficult topic of the year: SEM. I outlined how we can use SEM to run (almost) all the designs we've covered this year. SEM can even be used to create new analyses such as PCA with error, or regression with latent variables, or ANOVA on PCA-like factors, regression on factors with clustering, etc. I ended these notes by showing how SEM connects to the relatively complicated technique of multilevel modeling and latent growth modeling, which ties back to the most elementary of all statistical tests—the paired t test, which is just the basic ANOVA model with a grand mean, a fixed effect α for the two times, a random effect π for subject variability treated as a blocking factor, and an error term ϵ.

Hope you enjoyed the journey. May all your results be statistically significant, may your studies be sufficiently powered and, most importantly, may you have significant hypotheses to test!

The End

Appendix 1: Amos

Amos is a program that runs SEM. It is common at the University of Michigan because there is a site license for it. SPSS bought AMOS a few years ago, so the interface between AMOS and SPSS is relatively seamless. Eventually, IBM bought SPSS and all of its subprograms like Amos.

There are plenty of tutorials for AMOS on the web. Most rely on the graphical interface where the user enters circles, squares, arrows and double headed arrows. That is fine for the simplest of models, but for more realistic models the syntax is more efficient. The syntax is a little cumbersome (written in visual basic style), but it can do most things. Unfortunately, some of Amos' bells and whistles are not available through syntax file.

Here is the AMOS syntax. AMOS has a nice feature that it is possible to draw the SEM diagram on the screen to build the model (rather than entering syntax) but you really need to know what you are doing since there are constraints that need to be set and such. Best to start with syntax and then when you understand the program, work your way over to the graphical interface. The syntax though is not easy to read; it is possible that the syntax has been simplified and I missed the memo.

```
#Region "Header"
Imports System
Imports System.Diagnostics
Imports Microsoft.VisualBasic
Imports AmosEngineLib
Imports AmosGraphics
Imports AmosEngineLib.AmosEngine.TMatrixID
Imports PBayes
#End Region

Module MainModule
Public Sub Main()
Dim Sem As AmosEngineLib.AmosEngine=New AmosEngineLib.AmosEngine
Sem.TextOutput()
Sem.Standardized()
Sem.Smc()

Sem.BeginGroup ("f:\rich\teach\multstat\unixfiles\lectnotes\sem\rg1.sav")

     Sem.AStructure ("x1A   = (1) F1 + (1) err_x1A")
     Sem.AStructure ("x1B   =     F1 + (1) err_x1B")
     Sem.AStructure ("x1C   =     F1 + (1) err_x1C")
```

```
     Sem.AStructure ("x2A   = (1) F2 + (1) err_x2A")
     Sem.AStructure ("x2B   =     F2 + (1) err_x2B")
     Sem.AStructure ("x2C   =     F2 + (1) err_x2C")

     Sem.AStructure ("err_x1A (ex1A)")
     Sem.AStructure ("err_x1B (ex1B)")
     Sem.AStructure ("err_x1C (ex1C)")
     Sem.AStructure ("err_x2A (ex2A)")
     Sem.AStructure ("err_x2B (ex2B)")
     Sem.AStructure ("err_x2C (ex2C)")
     Sem.AStructure ("F1 (VF1)")
     Sem.AStructure ("F2 (VF2)")
Sem.Dispose()
End Sub
End Module
```

Let's walk through this syntax. The first few lines determine some of the output, such as setting text output and printing the standardized solution as well as the raw solution. The Smc piece adds the reliabilities to the output. The file containing the SPSS data is "rg1.sav". The next block of output sets up the six regression equations. For each observed variable we define a factor as a predictor and set up measurement error. Then we define the six errors of measurement and the variances of the two factors are set to 1. The correlation between the two factors is set to 0.

The output is presented in Figure 13-4. The output includes the six regression equations equations, with betas and their individual tests, the standarized betas, the six error variances, tests and goodness of fit measures for the entire model (all six regressions simultaneously fit and tested to the covariance matrix), and finally the reliabilities of each variable.

One may be interested in testing the correlation between the two factors. You run the same syntax except change the one line that fixes the correlation to 0 so that the program estimates the correlation. The one line of syntax is changed to the following where "corr" is an arbitrary label I assign to that parameter:

```
     Sem.AStructure ("F1 <> F2 (corr)")
```

After running this I get new output which I present only snippets in Figure 13-5. The covariance between the two factors is .206 and is statistically significant.

Figure 13-4: Two factor model with six observed variables. Factor variances set to 1.

Regression Weights: (Group number 1 - Model 1)

		Estimate	S.E.	C.R.	P	Label
x1A <---	F1	1.03924	.08134	12.77722	***	beta1
x1B <---	F1	1.26835	.10563	12.00761	***	beta2
x1C <---	F1	.87597	.08440	10.37900	***	beta3
x2A <---	F2	.98133	.08333	11.77677	***	beta4
x2B <---	F2	1.03091	.07877	13.08756	***	beta5
x2C <---	F2	1.39521	.10253	13.60817	***	beta6

Standardized Regression Weights: (Group number 1 - Model 1)

		Estimate
X1A <---	F1	.928
X1B <---	F1	.891
X1C <---	F1	.806
X2A <---	F2	.867
X2B <---	F2	.926
X2C <---	F2	.947

Variances: (Group number 1 - Model 1)

	Estimate	S.E.	C.R.	P	Label
F1	1.000				
F2	1.000				
err_x1A	.175	.056	3.121	.002	ex1A
err_x1B	.416	.093	4.449	***	ex1B
err_x1C	.413	.065	6.377	***	ex1C
err_x2A	.318	.051	6.269	***	ex2A
err_x2B	.177	.040	4.444	***	ex2B
err_x2C	.223	.066	3.363	***	

CMIN

Model	NPAR	CMIN	DF	P	CMIN/DF
Default model	12	13.724	9	.132	1.525
Saturated model	21	.000	0		
Independence model	6	576.533	15	.000	38.436

RMR, GFI

Model	RMR	GFI	AGFI	PGFI
Default model	.172	.965	.918	.413
Saturated model	.000	1.000		
Independence model	.669	.423	.193	.302

RMSEA

Model	RMSEA	LO 90	HI 90	PCLOSE
Default model	.066	.000	.133	.306
Independence model	.561	.522	.601	.000

Squared Multiple Correlations: (Group number 1 - Model 1)

	Estimate
x2C	.89717
x2B	.85708
x2A	.75187
x1C	.65032
x1B	.79467
x1A	.86063

Figure 13-5: Two factor model with six observed variables. Factor variances set to 1 and are allowed to be correlated.

Covariances: (Group number 1 - Model 1)

	Estimate	S.E.	C.R.	P	Label
F1 <--> F2	.20600	.09424	2.18597	.02882	corr

CMIN

Model	NPAR	CMIN	DF	P	CMIN/DF
Default model	13	9.25471	8	.32127	1.15684
Saturated model	21	.00000	0		
Independence model	6	576.53262	15	.00000	38.43551

RMR, GFI

Model	RMR	GFI	AGFI	PGFI
Default model	.05633	.97575	.93636	.37172
Saturated model	.00000	1.00000		
Independence model	.66869	.42338	.19273	.30241

Appendix 2: R

Cronbach's α

There are two packages (coefficientalpha and psy) that offer versions of Cronbach's α.

SEM

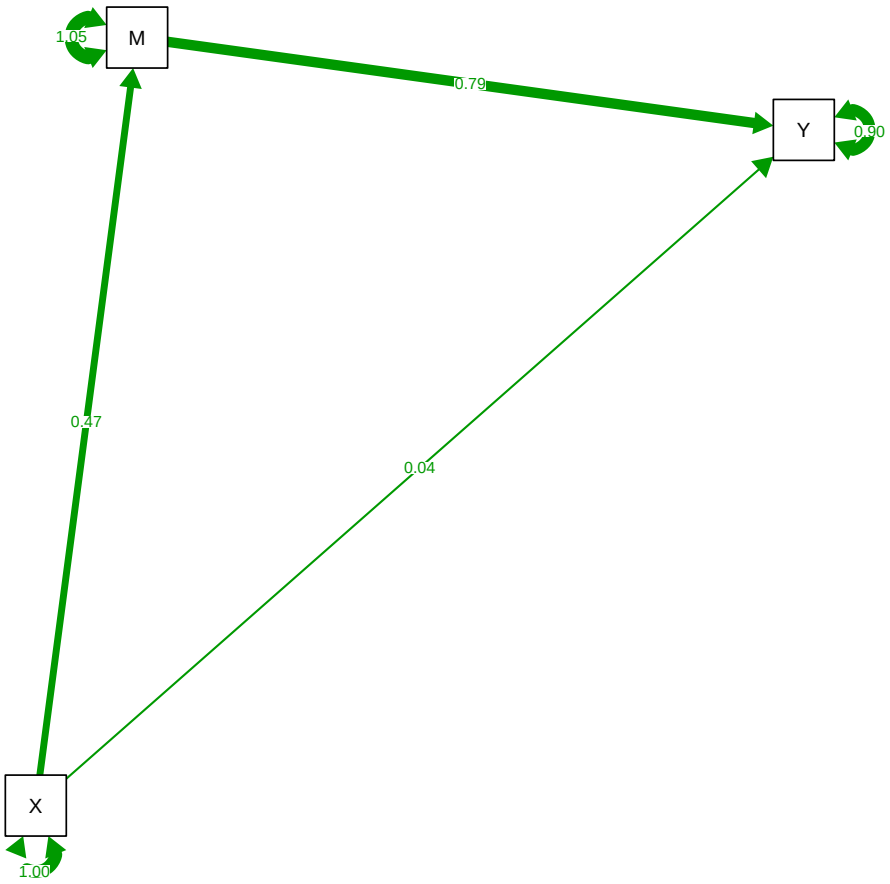
My preferred package for running SEM in R is lavaan. The syntax is very similar to Mplus so it is easy to go back an forth. There is an older package called sem that is pretty good too. For those who want to get very involved in SEM possibly even doing your own simulations and writing more advanced code to develop new methods, then the package openMX would be relevant.

Mediation tests can be worked into SEM programs. The lavaan package has a nice way of testing mediation including bootstrapping the indirect effect as a product term. Here is an example from the lavaan package documentation. One defines the linear equations, and then defines new parameters, say, 'ab' and 'total' that are functions of the estimated parameters. The default in lavaan is to estimate the standard errors of these derived parameters using approximations (see Gonzalez and Griffin, 2001, for an explanation about Wald tests), but one can specify a bootstrap version as well, which is illustrated below.

```
set.seed(1234)
X <- rnorm(100)
M <- 0.5 * X + rnorm(100)
Y <- 0.7 * M + rnorm(100)
Data <- data.frame(X = X, Y = Y, M = M)
model <- " # direct effect
             Y ~ c*X
           # mediator
             M ~ a*X
             Y ~ b*M
           # indirect effect (a*b)
             ab := a*b
           # total effect
             total := c + (a*b)
         "
fit <- sem(model, data = Data)
summary(fit)


## lavaan 0.6.17 ended normally after 1 iteration
##
```

```
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                         5
##
##   Number of observations                           100
##
## Model Test User Model:
##
##   Test statistic                                 0.000
##   Degrees of freedom                                 0
##
## Parameter Estimates:
##
##   Standard errors                             Standard
##   Information                                 Expected
##   Information saturated (h1) model          Structured
##
## Regressions:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   Y ~
##     X          (c)    0.036    0.104    0.348    0.728
##   M ~
##     X          (a)    0.474    0.103    4.613    0.000
##   Y ~
##     M          (b)    0.788    0.092    8.539    0.000
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##    .Y                 0.898    0.127    7.071    0.000
##    .M                 1.054    0.149    7.071    0.000
##
## Defined Parameters:
##                    Estimate  Std.Err  z-value  P(>|z|)
##     ab                0.374    0.092    4.059    0.000
##     total             0.410    0.125    3.287    0.001
```

```
semPaths(fit, what = "par", fade = FALSE, layout = "spring")
```



Lavaan can even get bootstrap standard errors; the implementation is relatively slow however. The standard errors are different from the previous output, hence the different z-values and the corresponding p-values. Because mediation is testing the product of two parameters (the path "a times b") and standard errors for products of random variables are more complicated, the convention is to use bootstrapping to estimate the standard errors in mediation.

```
fit <- sem(model, data = Data, se = "bootstrap")
summary(fit)


## lavaan 0.6.17 ended normally after 1 iteration
##
```

```
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                         5
##
##   Number of observations                           100
##
## Model Test User Model:
##
##   Test statistic                                 0.000
##   Degrees of freedom                                 0
##
## Parameter Estimates:
##
##   Standard errors                             Bootstrap
##   Number of requested bootstrap draws             1000
##   Number of successful bootstrap draws            1000
##
## Regressions:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   Y ~
##     X          (c)    0.036    0.112    0.325    0.745
##   M ~
##     X          (a)    0.474    0.099    4.807    0.000
##   Y ~
##     M          (b)    0.788    0.089    8.874    0.000
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##    .Y                 0.898    0.152    5.912    0.000
##    .M                 1.054    0.162    6.525    0.000
##
## Defined Parameters:
##                    Estimate  Std.Err  z-value  P(>|z|)
##     ab                0.374    0.084    4.437    0.000
##     total             0.410    0.131    3.131    0.002
```

To see confidence intervals of parameters use this command

```
parameterEstimates(fit)


##     lhs op     rhs label   est    se     z pvalue ci.lower
## 1     Y  ~       X     c 0.036 0.112 0.325  0.745   -0.156
## 2     M  ~       X     a 0.474 0.099 4.807  0.000    0.283
## 3     Y  ~       M     b 0.788 0.089 8.874  0.000    0.602
```

```
## 4     Y ~~       Y       0.898 0.152 5.912  0.000    0.605
## 5     M ~~       M       1.054 0.162 6.525  0.000    0.733
## 6     X ~~       X       0.999 0.000    NA     NA    0.999
## 7    ab :=     a*b    ab 0.374 0.084 4.437  0.000    0.218
## 8 total := c+(a*b) total 0.410 0.131 3.131  0.002    0.168
##   ci.upper
## 1    0.275
## 2    0.666
## 3    0.945
## 4    1.200
## 5    1.381
## 6    0.999
## 7    0.548
## 8    0.675
```

One can combine a factor model to assess measurement error and also a mediation model. Here is a simplified version of the classic example from Bollen's SEM textbook.

```
model <- "
  # latent variable definitions
     ind60 =~ x1 + x2 + x3
     dem60 =~ y1 + a*y2 + b*y3 + c*y4
     dem65 =~ y5 + a*y6 + b*y7 + c*y8

  # regressions
    dem60 ~ apath* ind60
    dem65 ~ cpath*ind60 + bpath*dem60

 # define indirect effect (a*b)
             ab := apath*bpath

 # define total effect
             total := cpath + (apath*bpath)
"

fit <- sem(model, data = PoliticalDemocracy)
summary(fit, fit.measures = TRUE)


## lavaan 0.6.17 ended normally after 40 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                        25
##   Number of equality constraints                     3
```

```
##
##   Number of observations                            75
##
## Model Test User Model:
##
##   Test statistic                                74.618
##   Degrees of freedom                               44
##   P-value (Chi-square)                          0.003
##
## Model Test Baseline Model:
##
##   Test statistic                               730.654
##   Degrees of freedom                                55
##   P-value                                        0.000
##
## User Model versus Baseline Model:
##
##   Comparative Fit Index (CFI)                    0.955
##   Tucker-Lewis Index (TLI)                       0.943
##
## Loglikelihood and Information Criteria:
##
##   Loglikelihood user model (H0)              -1566.037
##   Loglikelihood unrestricted model (H1)      -1528.728
##
##   Akaike (AIC)                                3176.075
##   Bayesian (BIC)                              3227.059
##   Sample-size adjusted Bayesian (SABIC)       3157.721
##
## Root Mean Square Error of Approximation:
##
##   RMSEA                                          0.096
##   90 Percent confidence interval - lower         0.057
##   90 Percent confidence interval - upper         0.133
##   P-value H_0: RMSEA <= 0.050                    0.031
##   P-value H_0: RMSEA >= 0.080                    0.775
##
## Standardized Root Mean Square Residual:
##
##   SRMR                                           0.065
##
## Parameter Estimates:
##
##   Standard errors                             Standard
```

```
##   Information                                 Expected
##   Information saturated (h1) model          Structured
##
## Latent Variables:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   ind60 =~
##     x1                1.000
##     x2                2.181    0.139   15.716    0.000
##     x3                1.819    0.152   11.958    0.000
##   dem60 =~
##     y1                1.000
##     y2         (a)    1.287    0.118   10.875    0.000
##     y3         (b)    1.174    0.107   10.930    0.000
##     y4         (c)    1.299    0.102   12.729    0.000
##   dem65 =~
##     y5                1.000
##     y6         (a)    1.287    0.118   10.875    0.000
##     y7         (b)    1.174    0.107   10.930    0.000
##     y8         (c)    1.299    0.102   12.729    0.000
##
## Regressions:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   dem60 ~
##     ind60   (apth)    1.463    0.383    3.822    0.000
##   dem65 ~
##     ind60   (cpth)    0.464    0.223    2.082    0.037
##     dem60   (bpth)    0.887    0.074   12.053    0.000
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##    .x1                0.082    0.020    4.179    0.000
##    .x2                0.119    0.070    1.693    0.090
##    .x3                0.467    0.090    5.174    0.000
##    .y1                1.921    0.389    4.944    0.000
##    .y2                6.620    1.186    5.582    0.000
##    .y3                5.321    0.957    5.562    0.000
##    .y4                2.917    0.607    4.803    0.000
##    .y5                2.379    0.446    5.335    0.000
##    .y6                4.343    0.803    5.408    0.000
##    .y7                3.604    0.667    5.406    0.000
##    .y8                2.936    0.586    5.013    0.000
##     ind60             0.448    0.087    5.170    0.000
##    .dem60             3.794    0.811    4.676    0.000
##    .dem65             0.120    0.208    0.577    0.564
```

```
##
## Defined Parameters:
##                   Estimate  Std.Err  z-value  P(>|z|)
##     ab               1.298    0.357    3.637    0.000
##     total            1.761    0.361    4.885    0.000
```

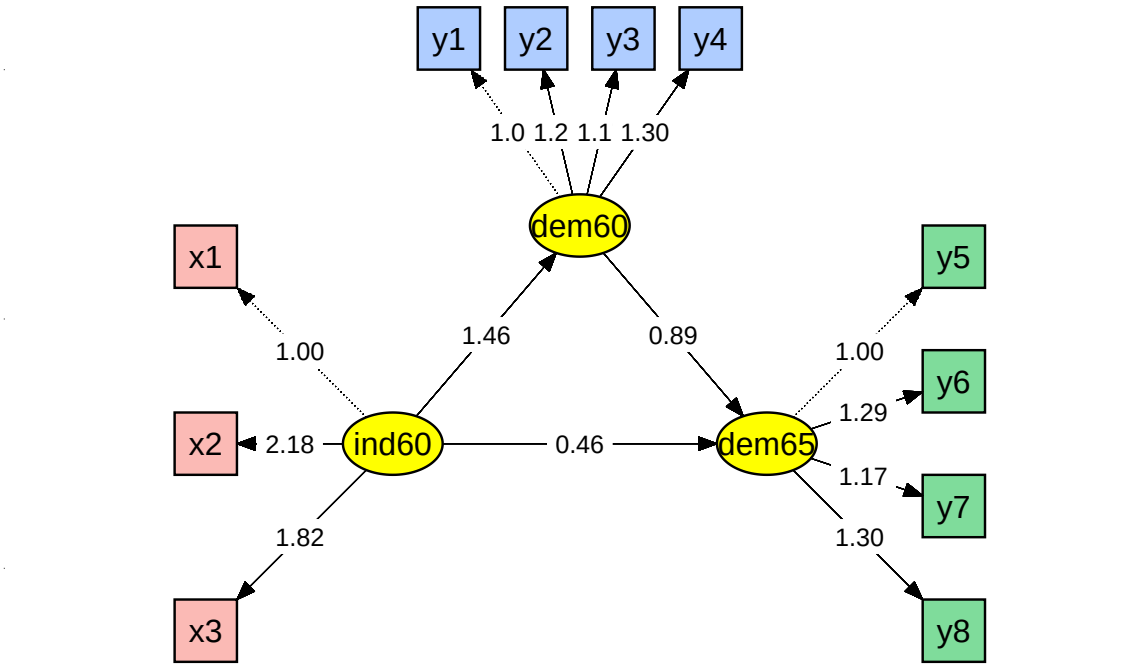
```
parameterEstimates(fit)
```

```
##      lhs op                 rhs label   est    se      z
## 1  ind60 =~                  x1       1.000 0.000     NA
## 2  ind60 =~                  x2       2.181 0.139 15.716
## 3  ind60 =~                  x3       1.819 0.152 11.958
## 4  dem60 =~                  y1       1.000 0.000     NA
## 5  dem60 =~                  y2     a 1.287 0.118 10.875
## 6  dem60 =~                  y3     b 1.174 0.107 10.930
## 7  dem60 =~                  y4     c 1.299 0.102 12.729
## 8  dem65 =~                  y5       1.000 0.000     NA
## 9  dem65 =~                  y6     a 1.287 0.118 10.875
## 10 dem65 =~                  y7     b 1.174 0.107 10.930
## 11 dem65 =~                  y8     c 1.299 0.102 12.729
## 12 dem60  ~                ind60 apath 1.463 0.383  3.822
## 13 dem65  ~                ind60 cpath 0.464 0.223  2.082
## 14 dem65  ~                dem60 bpath 0.887 0.074 12.053
## 15    x1 ~~                   x1       0.082 0.020  4.179
## 16    x2 ~~                   x2       0.119 0.070  1.693
## 17    x3 ~~                   x3       0.467 0.090  5.174
## 18    y1 ~~                   y1       1.921 0.389  4.944
## 19    y2 ~~                   y2       6.620 1.186  5.582
## 20    y3 ~~                   y3       5.321 0.957  5.562
## 21    y4 ~~                   y4       2.917 0.607  4.803
## 22    y5 ~~                   y5       2.379 0.446  5.335
## 23    y6 ~~                   y6       4.343 0.803  5.408
## 24    y7 ~~                   y7       3.604 0.667  5.406
## 25    y8 ~~                   y8       2.936 0.586  5.013
## 26 ind60 ~~                ind60       0.448 0.087  5.170
## 27 dem60 ~~                dem60       3.794 0.811  4.676
## 28 dem65 ~~                dem65       0.120 0.208  0.577
## 29    ab :=          apath*bpath    ab 1.298 0.357  3.637
## 30 total := cpath+(apath*bpath) total 1.761 0.361  4.885
##    pvalue ci.lower ci.upper
## 1      NA    1.000    1.000
## 2   0.000    1.909    2.453
## 3   0.000    1.521    2.117
## 4      NA    1.000    1.000
```

```
## 5   0.000    1.055    1.520
## 6   0.000    0.964    1.385
## 7   0.000    1.099    1.499
## 8      NA    1.000    1.000
## 9   0.000    1.055    1.520
## 10  0.000    0.964    1.385
## 11  0.000    1.099    1.499
## 12  0.000    0.713    2.213
## 13  0.037    0.027    0.900
## 14  0.000    0.743    1.031
## 15  0.000    0.043    0.120
## 16  0.090   -0.019    0.256
## 17  0.000    0.290    0.644
## 18  0.000    1.159    2.682
## 19  0.000    4.296    8.944
## 20  0.000    3.446    7.196
## 21  0.000    1.727    4.107
## 22  0.000    1.505    3.252
## 23  0.000    2.769    5.917
## 24  0.000    2.297    4.910
## 25  0.000    1.788    4.084
## 26  0.000    0.278    0.618
## 27  0.000    2.203    5.384
## 28  0.564   -0.287    0.527
## 29  0.000    0.598    1.997
## 30  0.000    1.055    2.468
```

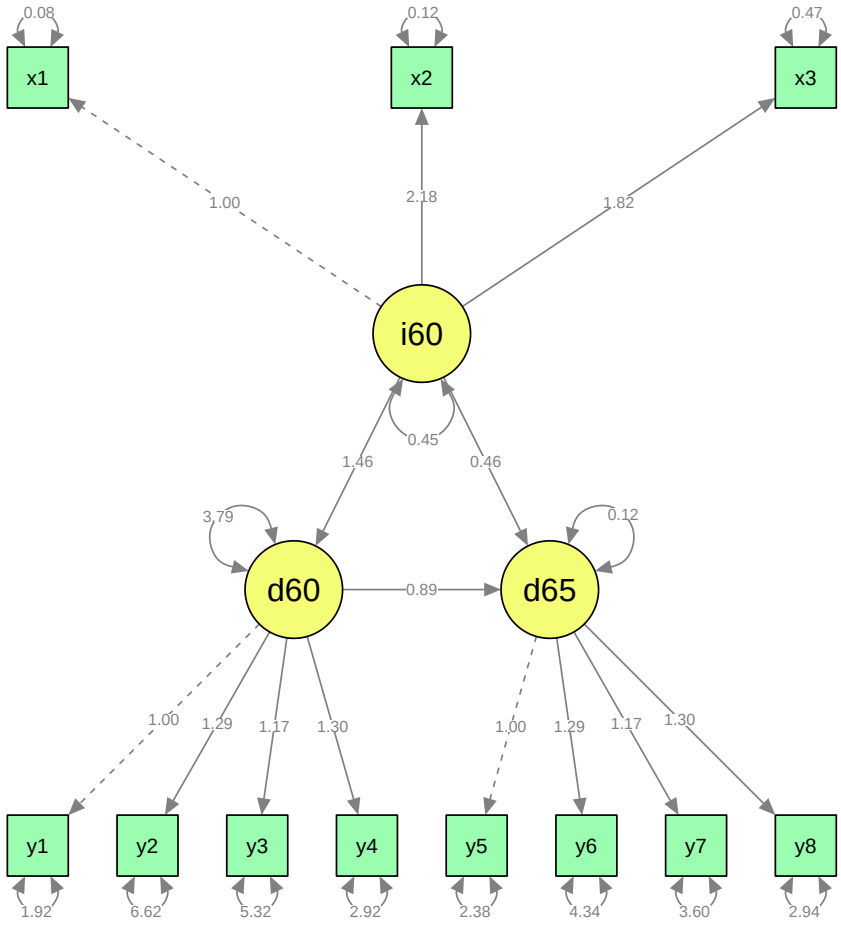
There are also a few packages specifically for mediation, including the mediation package that has a Bayesian implementation and the semMediation package that adds functionality to a lavaan model (semMediation though seems to not be supported anymore and depends on packages that are also no longer supported). Here is a nice plotting command that is included in the semMediation package.

```
# install package devtools install.packages('devtools') the
# double colon :: is to call a function in a package
# without loading complete library
# devtools::install_github('davidgohel/ReporteRsjars')
# devtools::install_github('davidgohel/ReporteRs')
# devtools::install_github('guhjy/semMediation')
semMediation::mediationPlot(fit, base_family = "sans", whatLabels = "est")
```



You can also use the semPlot package to plot the lavaan output.

```
semPaths(fit, whatLabels = "par", what = "path", color = list(lat = rgb(245,
    253, 118, maxColorValue = 255), man = rgb(155, 253, 175,
    maxColorValue = 255)))
```

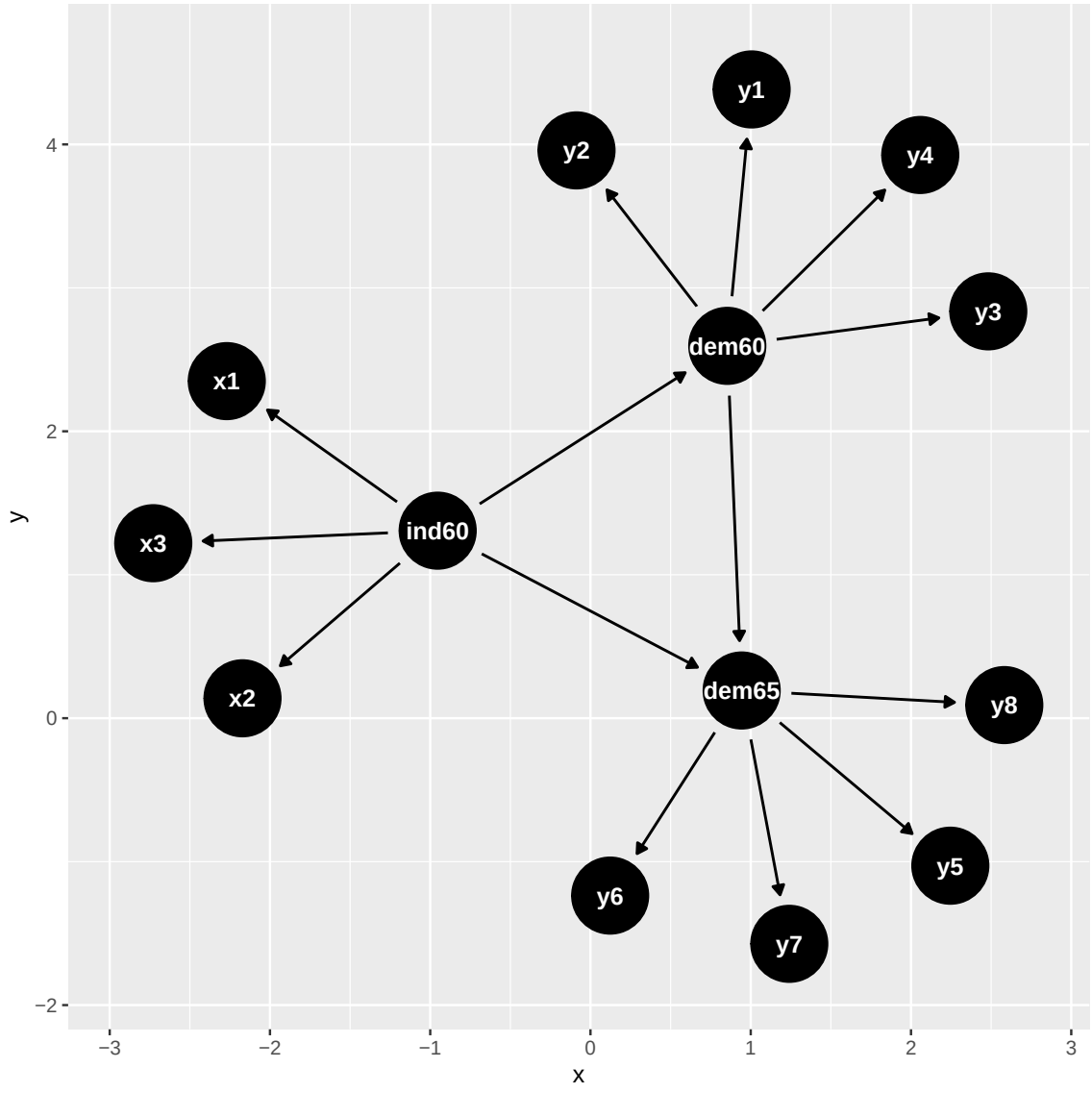


The figure though may seem confusing because the double-headed arrow representing the variance of the latent variable i60 covers the single-headed arrows out of i60 making them look like double-headed arrows.

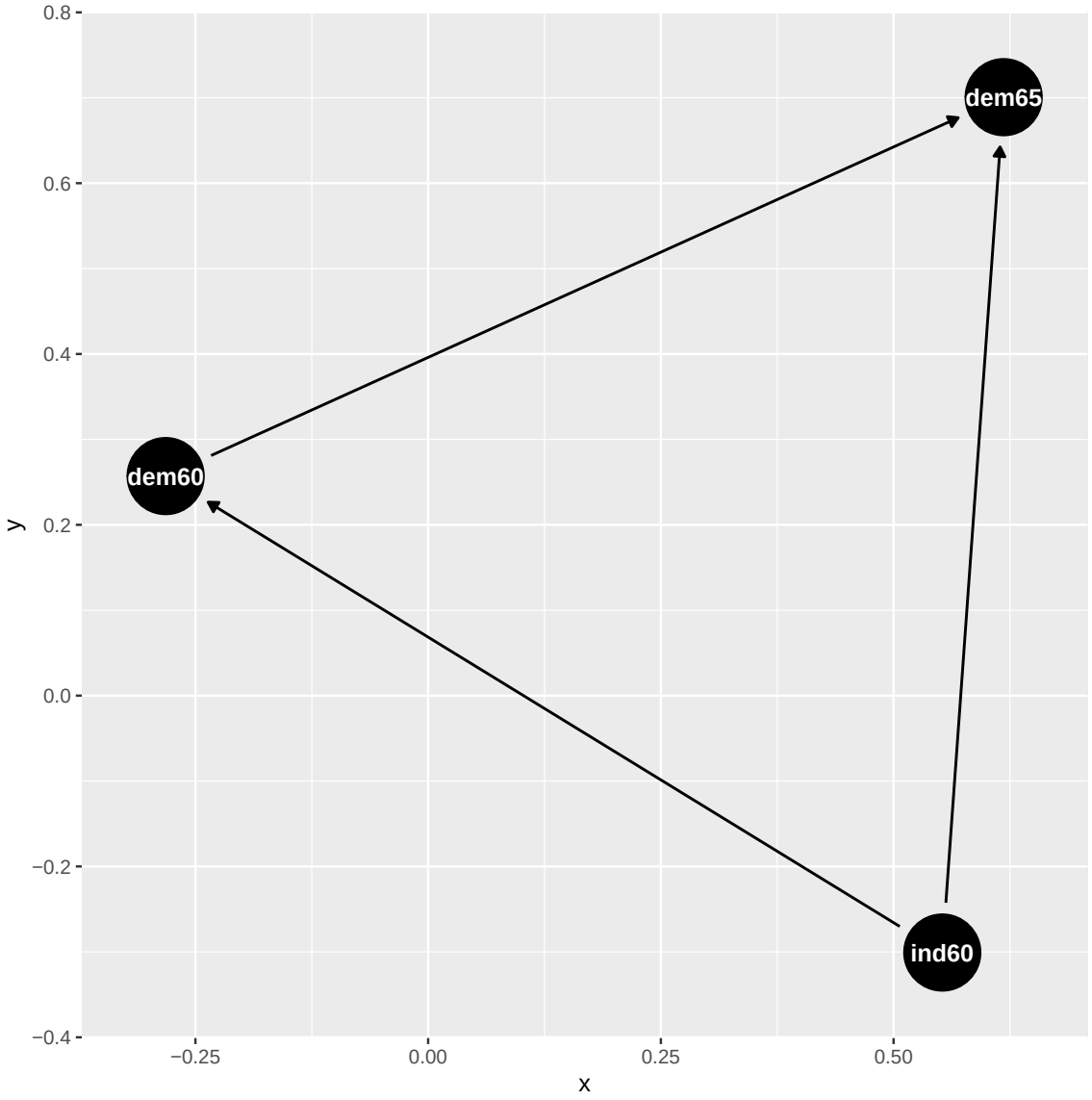
The reason it is a good idea to merge factor analysis when testing mediation is that the mediation is conducted at the factor level, meaning the measurement error has been accounted for through the measurement part of the model (i.e. the residual variances). This eliminates the concern we saw earlier in these lecture notes of how measurement error in one or more predictors can create issues in a multiple regression.

Directed Acyclic Graphs (DAGS)

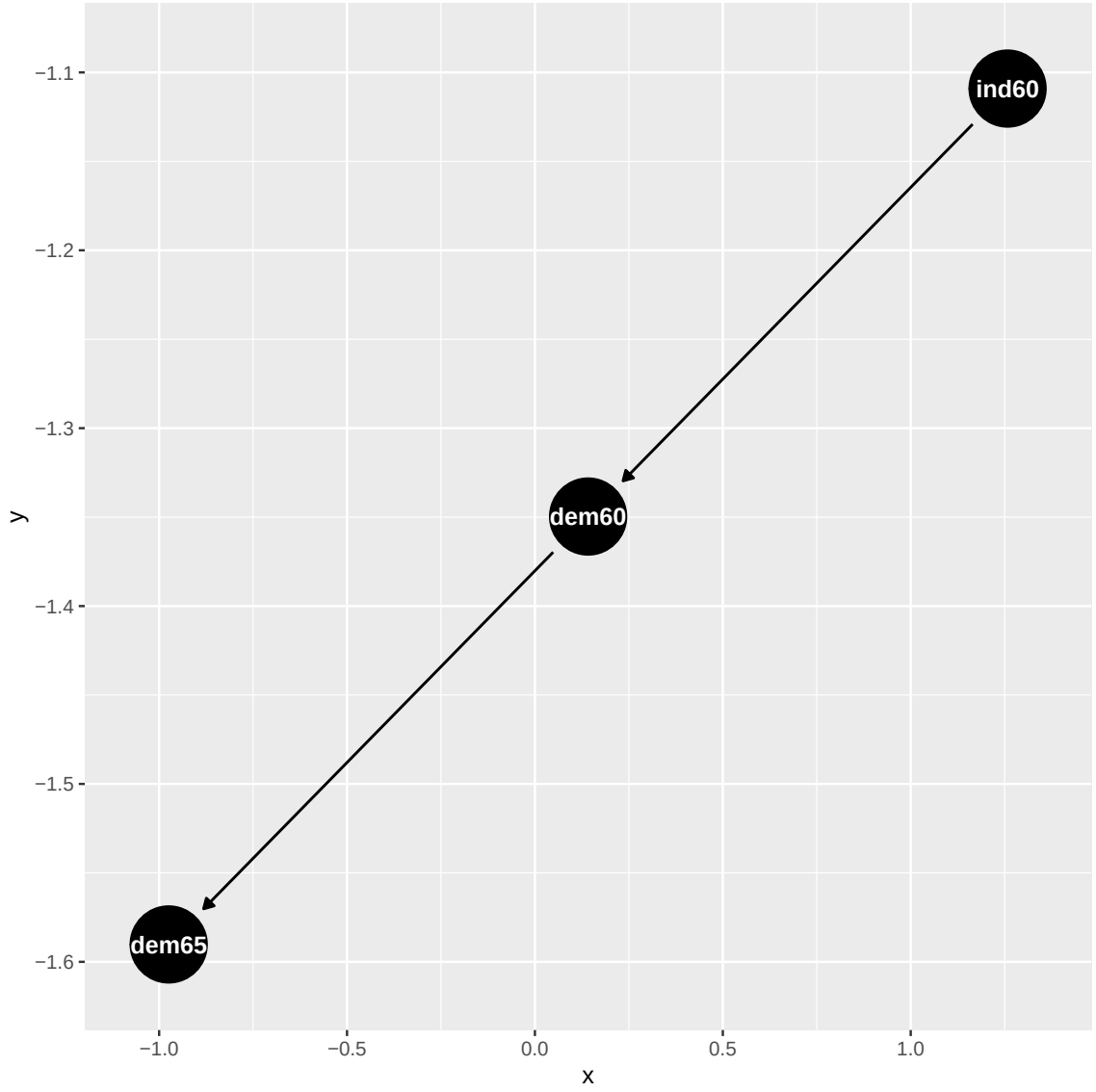
```
library(dagitty)
library(ggdag)
library(ggplot2)
out.dag <- dagitty(lavaanToGraph(model))
ggdag(out.dag)
```




```
ggdag(structuralPart(out.dag))
```




```
impliedConditionalIndependencies(out.dag)

# try model without direct path
model2 <- "
    dem60 ~ ind60
    dem65 ~ dem60
"
out.dag2 <- dagitty(lavaanToGraph(model2))
ggdag(out.dag2)
```




```
impliedConditionalIndependencies(out.dag2)


## dm65 _||_ in60 | dm60
```

The latter model implies that dem65 and ind60 will be independent (not correlated) when the mediator dem60 is controlled (i.e., the model implies that the beta from ind60 to dem65 is 0).

Instrumental variables

In addition to the packages already mentioned for conducting instrumental variables, the sem package has a function to compute instrumental variables analysis. Econometricians are now beginning to use R so there are more instrumental variable tools available (STATA is the economists' SPSS).

Also, check out the package MIIVsem that takes SEM models (like lavaan syntax) and expresses them in terms of instrumental variables. Here is an example using the mediation model and data described above under SEM. Bollen is both an initial developer of SEM with a classic textbook and now a proponent of running SEM through an instrumental variables approach. I illustrate with his MIIVsem package.

```
library(MIIVsem)
model <- " # outcome regression
             Y ~ c*X + b*M
           # mediator regression
             M ~ a*X
         "

miivs(model)


## Model Equation Information
##
##  LHS RHS    MIIVs
##  Y   X, M   M, X
##  M   X      X


print(miive(model, Data))


## MIIVsem (0.5.8) results
##
## Number of observations                                          100
## Number of equations                                               2
## Estimator                                                    MIIV-
2SLS
## Standard Errors                                             standard
## Missing                                                     listwise
##
##
## Parameter Estimates:
##
##
```

```
## STRUCTURAL COEFFICIENTS:
##                    Estimate  Std.Err  z-value
##   M ~
##     X                 0.474    0.103    4.613
##   Y ~
##     X                 0.036    0.104    0.348
##     M                 0.788    0.092    8.539
##   P(>|z|)   Sargan   df   P(Chi)
##
##     0.000
##
##     0.728
##     0.000
##
## INTERCEPTS:
##                    Estimate  Std.Err  z-value
##     M                 0.037    0.104    0.358
##     Y                 0.164    0.096    1.704
##   P(>|z|)
##     0.721
##     0.088
```

Here is the instrumental variables approach to the factor analytic mediation model I presented earlier.

```
model <- "
  # latent variable definitions
     ind60 =~ x1 + x2 + x3
     dem60 =~ y1 + a*y2 + b*y3 + c*y4
     dem65 =~ y5 + a*y6 + b*y7 + c*y8

  # regressions
    dem60 ~ apath* ind60
    dem65 ~ cpath*ind60 + bpath*dem60
"

miivs(model)


## Model Equation Information
##
##  LHS  RHS      MIIVs
##  x2   x1       x3, y1, y2, y3, y4, y5, y6, y7, y8
##  x3   x1       x2, y1, y2, y3, y4, y5, y6, y7, y8
##  y2   y1       x1, x2, x3, y3, y4, y5, y6, y7, y8
```

```
##  y3   y1       x1, x2, x3, y2, y4, y5, y6, y7, y8
##  y4   y1       x1, x2, x3, y2, y3, y5, y6, y7, y8
##  y6   y5       x1, x2, x3, y1, y2, y3, y4, y7, y8
##  y7   y5       x1, x2, x3, y1, y2, y3, y4, y6, y8
##  y8   y5       x1, x2, x3, y1, y2, y3, y4, y6, y7
##  y1   x1       x2, x3
##  y5   x1, y1   x2, x3, y2, y3, y4

print(miive(model, data = PoliticalDemocracy))

## MIIVsem (0.5.8) results
##
## Number of observations                                          75
## Number of equations                                             10
## Estimator                                                  MIIV-
2SLS
## Standard Errors                                          standard
## Missing                                                  listwise
##
##
## Parameter Estimates:
##
##
## STRUCTURAL COEFFICIENTS:
##                    Estimate  Std.Err  z-value
##   dem60 =~
##     y1                1.000
##     y2                1.159    0.116   10.031
##     y3                1.058    0.094   11.269
##     y4                1.164    0.097   11.984
##   dem65 =~
##     y5                1.000
##     y6                1.159    0.116   10.031
##     y7                1.058    0.094   11.269
##     y8                1.164    0.097   11.984
##   ind60 =~
##     x1                1.000
##     x2                2.078    0.128   16.171
##     x3                1.751    0.149   11.782
##   P(>|z|)   Sargan   df   P(Chi)
##
##
##     0.000   19.516    8    0.012
##     0.000   10.169    8    0.253
```

```
##     0.000   15.024    8    0.059
##
##
##     0.000   19.882    8    0.011
##     0.000   14.631    8    0.067
##     0.000   15.154    8    0.056
##
##
##     0.000    8.301    8    0.405
##     0.000    8.738    8    0.365
##
##   dem60 ~
##     ind60             1.261    0.426    2.962
##   dem65 ~
##     ind60             1.123    0.312    3.598
##     dem60             0.724    0.101    7.140
##
##
##     0.003    0.503    1    0.478
##
##     0.000    0.801    3    0.849
##     0.000
##
## INTERCEPTS:
##                    Estimate  Std.Err  z-value
##     dem60            -0.909    2.170   -0.419
##     dem65            -4.499    1.424   -3.160
##     x1                0.000
##     x2               -5.711    0.654   -8.727
##     x3               -5.292    0.758   -6.985
##     y1                0.000
##     y2               -2.077    0.731   -2.841
##     y3                0.783    0.586    1.336
##     y4               -1.908    0.605   -3.156
##     y5                0.000
##     y6               -2.975    0.686   -4.336
##     y7                0.764    0.559    1.365
##     y8               -1.935    0.588   -3.292
##   P(>|z|)
##     0.675
##     0.002
##
##     0.000
##     0.000
```

```
##
##      0.005
##      0.181
##      0.002
##
##      0.000
##      0.172
##      0.001
```