Introduction to Structural Equation Modeling: The General Model

Henrik Kenneth Andersen

2021-10-08

Introduction

The 'General' or 'Full' SEM combines regression and/or mediation with confirmatory factor analysis to model the 'structural' relations between the latent variables.

Measurement error and attenuation bias

What if the observed variables are not measured perfectly? Then what we observe, call them $\tilde{x}$ and $\tilde{y}$ are composites of the true score we are after, i.e., x and y, plus an additive measurement error portion:

$$\begin{aligned}\tilde{x} &= x + \upsilon,\\ \tilde{y} &= y + \nu.\end{aligned}$$

How does this affect our model? Well, first notice that measurement error in the dependent variable is typically less of a serious problem than measurement error in the independent variables. Let us assume mean-centered variables so that we can ignore the intercept, and consider the following simple bivariate equation:

$$y = \beta x + \varepsilon$$

if y is measured imperfectly and what we observe is $\tilde{y} = y + \nu$, then we can rewrite the equation as:

$$\begin{aligned}(\tilde{y} - \nu) &= \beta x + \varepsilon\\ \tilde{y} &= \beta x + \varepsilon + \nu.\end{aligned}$$

The measurement error in y just gets added to the regression error. As long as ν is uncorrelated with x, then the regression coefficient will be unbiased (Pischke 2007; Wooldridge 2009). However, this will increase the error variance and thus make the estimates less precise, i.e., higher standard error, lower R^2.

We will look at the effect of measurement error in the dependent variable using an example shortly. For now though, let us be safe in the knowledge that the coefficient of interest is likely unbiased, and concentrate on the more serious problem of error in the independent variable.

The intuition behind the problem of measurement error in the independent variable(s) can be explained as follows. Take $\tilde{x} = x + \upsilon$ and substitute this into the equation for y:

$$\begin{aligned}y &= \beta x + \varepsilon\\ &= \beta(\tilde{x} - \upsilon) + \varepsilon\\ &= \beta\tilde{x} + (\varepsilon - \beta\upsilon).\end{aligned}$$

Since $\tilde{x}$ is obviously correlated with υ (unless the variance of υ is so small so that the correlation is essentially negligible), then the composite error in this regression is also correlated with the independent variable and thus the estimated coefficient of β will be biased.

Simulated examples

Let us simulate some data to demonstrate the problem of measurement error and attenuation bias. We need to use simulation because we generally do not know with observed variables whether they are measured with error and what portion of their variance is due to it.

```
library(lavaan)

# Set seed for replicability
set.seed(1234)

# Set large sample size to minimize sampling error
n <- 10000
# Set population effect
beta <- 0.5

# Make the independent variable
x <- rnorm(n = n, mean = 0, sd = 1)
# Dependent variable
y <- beta * x + rnorm(n = n, mean = 0, sd = 1)

# Put together into dataframe
df <- data.frame(x, y)

# Run the simple lin. reg. model in SEM
m1 <- '
y ~ beta*x
x ~~ phi*x
y ~~ psi*y
'
m1.fit <- sem(model = m1, data = df)
summary(m1.fit)
```

```
## lavaan 0.6-8 ended normally after 11 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                         3
##
##   Number of observations                         10000
##
## 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       (beta)    0.511    0.010   50.161    0.000
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##     x         (phi)    0.975    0.014   70.711    0.000
##    .y         (psi)    1.012    0.014   70.711    0.000
```

From this, we see that the properly specified model returns a consistent and unbiased estimate of the effect with $\hat{\beta} = 0.511^{***}$ (0.010).

But now if we add error to x that is independent of y, the model including the predictor measured with error does not perform well, at all.

```
# Now add measurement error to x
xtilde <- x + rnorm(n = n, mean = 0, sd = 1)

# Add xtilde to dataframe
df <- data.frame(x, y, xtilde)

# Re-run the model with measurement error sullied independent variable
m2 <- '
y ~ beta*xtilde
xtilde ~~ phi*xtilde
y ~~ psi*y
'
m2.fit <- sem(model = m2, data = df)
summary(m2.fit)
```

```
## lavaan 0.6-8 ended normally after 12 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                         3
##
##   Number of observations                         10000
##
## 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 ~
##     xtilde  (beta)    0.252    0.008   32.857    0.000
##
## Variances:
```

```
##                    Estimate  Std.Err  z-value  P(>|z|)
##     xtilde   (phi)    1.944    0.027   70.711    0.000
##    .y        (psi)    1.143    0.016   70.711    0.000
```

Here, the effect is downward biased to a substantial degree, with $\hat{\beta}^{M2} = 0.252^{***}$ (0.008). This is because the measurement error in the predictor gets relegated to the error, which violates the assumption that the error is uncorrelated with the predictor.

Recall that we said measurement error in the dependent variable is less serious, because it will not bias estimates but will make them less precise. To demonstrate this, let us return to the first model where the independent variable is measured without error, but let us increase the unexplained variance in y (it already had an error variance, let us just make it bigger). We could imagine that our instrument was not very reliable and the amount of random error was increased.

```
# Increase the error variance in y
ytilde <- y + rnorm(n = n, mean = 0, sd = 3)

# Put ytilde into the dataframe
df <- data.frame(x, y, xtilde, ytilde)

# Re-run the model with measurement error sullied dependent variable
m3 <- '
ytilde ~ beta*x
x ~~ phi*x
ytilde ~~ psi*ytilde
'
m3.fit <- sem(model = m3, data = df)
summary(m3.fit)
```

```
## lavaan 0.6-8 ended normally after 16 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                         3
##
##   Number of observations                         10000
##
## 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|)
##   ytilde ~
##     x       (beta)    0.518    0.032   16.044    0.000
##
## Variances:
```

```
##                    Estimate  Std.Err  z-value  P(>|z|)
##     x        (phi)    0.975    0.014   70.711    0.000
##    .ytilde   (psi)   10.156    0.144   70.711    0.000
```

The regression coefficient is not identical to `m1.fit` simply due to sampling error, but the coefficient is still unbiased and consistent. However, the standard error has increased from 0.010 in `m1.fit` to 0.032 in `m3.fit`.

We can also look at the R^2 statistics using `lavInspect(model, "r2")`. In the first model, we were able to explain roughly 20% of the variance in y with the predictor x.

```
# R^2 for m1.fit
lavInspect(m1.fit, "r2")
```

```
##     y
## 0.201
```

Now, let us look at the model in which the dependent variable was less accurately measured:

```
# R^2 for m3.fit
lavInspect(m3.fit, "r2")
```

```
## ytilde
##  0.025
```

In model 3, even though the model is specified correctly and the coefficient is unbiased and consistent, we are no only able to explain about 2.5% of the variance of y.

For these reasons, measurement error in both the independent and dependent variables should be avoided if possible. SEM provides a framework for dealing with it.

Multiple indicator measurement models

We can think of the classical test theory equations above, i.e., observed score = true score + error in terms of latent variables. We say the 'true score' we are after is the score on the underlying latent variable, which is unobservable. The latent variable 'causes' the observed scores to some extent, but there is some amount of error that makes it so the observed score does not exactly equal the true score.

Let us say that we had multiple indicators that are meant to measure the same construct, e.g., an established measurement scale for some attitudinal or psychological construct. Then, just like in the CFA case, we would define so-called measurement models that express the observed variable in terms of the unobserved construct of interest and some error

$$x_k = \lambda_k^x \xi + \delta_k$$
$$y_k = \lambda_k^y \eta + \epsilon_k$$

where $k = 1, \ldots, K$. These equations say that the observed indicators, x_k, y_k are the result of the actual construct we are trying to measure, ξ, η, linked by factor loadings representing the extent to which the observed variable is influenced by the latent construct and some measurement error, δ_k, ϵ_k. This is exactly what we were doing in the CFA section, explaining the observed variable in terms of the latent variables and decomposing their variance into 'valid' and 'error' variance portions.

Once we have decomposed the variables, we set the structural relations between the measurement-error-free latent variables, with the assumption that the measurement error is independent of those latent variables.

Let us continue with the simulated example, because it should be very transparent that when our assumptions hold and we have good measures of the underlying latent construct, that we can avoid attenuation bias. Let us simulate three indicators for each the underlying latent exogenous and endogenous variables.

```
# Set seed for replicability
set.seed(987)

# Set large sample size to minimize sampling error
n <- 10000

# 'True' coefficient is 0.5
gamma <- 0.5

# Latent exogenous construct
xi <- rnorm(n = n, mean = 0, sd = 1)
# Latent endogenous construct
eta <- gamma * xi + rnorm(n = n, mean = 0, sd = 1)

# Measurements with error
x1 <- 0.8 * xi + rnorm(n = n, mean = 0, sd = 1)
x2 <- 0.7 * xi + rnorm(n = n, mean = 0, sd = 1)
x3 <- 0.9 * xi + rnorm(n = n, mean = 0, sd = 1)
y1 <- 0.7 * eta + rnorm(n = n, mean = 0, sd = 1)
y2 <- 0.7 * eta + rnorm(n = n, mean = 0, sd = 1)
y3 <- 0.8 * eta + rnorm(n = n, mean = 0, sd = 1)

# Dataframe of latent variables
dflat <- data.frame(xi, eta)
# Dataframe of observed variables
dfobs <- data.frame(x1, x2, x3, y1, y2, y3)

# 'Baseline' model (mbl)
mbl <- '
# Regression between true underlying variables
  eta ~ gamma*xi
# Variances
  xi  ~~ phi*xi
  eta ~~ psi*eta
'
mbl.fit <- sem(model = mbl, data = dflat)
summary(mbl.fit)
```

```
...
## Regressions:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   eta ~
##     xi      (gamm)    0.501    0.010   50.512    0.000
...
```

```
# 'Observed' model with one indicator per underlying construct
# Essentially assumption that error variance = 0 is wrong
mobs <- '
# Regression between observed indicators
```

```
  y1 ~ gamma*x1
# Variances
  x1 ~~ phi*x1
  y1 ~~ psi*y1
'
mobs.fit <- sem(model = mobs, data = dfobs)
summary(mobs.fit)
```

```
...
## Regressions:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   y1 ~
##     x1      (gamm)    0.173    0.010   17.945    0.000
...
```

```
# 'Error corrected' model (mec) with multiple indicator measurement models
mec <- '
# Measurement models
  xi1  =~ 1*x1 + l2*x2 + l3*x3
  eta1 =~ 1*y1 + l2*y2 + l3*y3
# Regression at latent variable-level
  eta1 ~ gamma*xi1
# Variances
  xi1  ~~ phi*xi1
  eta1 ~~ psi*eta1
# Measurement error
  x1 ~~ d1*x1
  x2 ~~ d2*x2
  x3 ~~ d3*x3
  y1 ~~ e1*y1
  y2 ~~ e2*y2
  y3 ~~ e3*y3
'
mec.fit <- sem(model = mec, data = dfobs)
summary(mec.fit)
```

```
...
## Regressions:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   eta1 ~
##     xi1     (gamm)    0.457    0.015   29.913    0.000
...
```

Obviously, the model that corrects for measurement error gets much closer to the true effect of $\gamma = 0.5$.

Note, however, that the ability for SEMs to correct for measurement error depends on the quality of the measurements. We can imagine a scenario in which the indicators are measured with minimal error. Then, the results of the SEM regression should closely match the hypothetical regression between the true underlying constructs. The poorer the measurement model, the less capable the model will be of identifying the true effect.

Take the following example, in which the factor loadings are all uniformly poor.

```
# --- Same model, but lower factor loadings
x1 <- 0.2 * xi + rnorm(n = n, mean = 0, sd = 1)
x2 <- 0.2 * xi + rnorm(n = n, mean = 0, sd = 1)
x3 <- 0.2 * xi + rnorm(n = n, mean = 0, sd = 1)
y1 <- 0.2 * eta + rnorm(n = n, mean = 0, sd = 1)
y2 <- 0.2 * eta + rnorm(n = n, mean = 0, sd = 1)
y3 <- 0.2 * eta + rnorm(n = n, mean = 0, sd = 1)

dfobs <- data.frame(x1, x2, x3, y1, y2, y3)

mec <- '
# Measurement models
  xi1  =~ 1*x1 + l2*x2 + l3*x3
  eta1 =~ 1*y1 + l2*y2 + l3*y3
# Regression at latent variable-level
  eta1 ~ gamma*xi1
# Variances
  xi1  ~~ phi*xi1
  eta1 ~~ psi*eta1
# Measurement error
  x1 ~~ d1*x1
  x2 ~~ d2*x2
  x3 ~~ d3*x3
  y1 ~~ e1*y1
  y2 ~~ e2*y2
  y3 ~~ e3*y3
'
mec.fit <- sem(model = mec, data = dfobs)
summary(mec.fit)
```

```
...
## Regressions:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   eta1 ~
##     xi1     (gamm)    0.774    0.178    4.339    0.000
...
```

Weak factor loadings, which signify low inter-item correlations, actually produces an *overcorrection* in which the regression coefficient(s) become inflated. Here, the estimated effect is much higher than the true effect of 0.5. For this reason, the full SEM requires that the measurement models are sound, where we would ideally have all factor loadings > 0.7 or so.

Another scenario arises when the error variances is very large. Obviously, even if the factor loadings are sound, if the amount of 'noise' in the measures is too large, we will not be able to recognize the systemic covariation as well. This will lead to downward biased estimates and low levels of explained variance as shown in the fictitious example below.

```
# --- Same model, but larger error variances

x1 <- 0.8 * xi + rnorm(n = n, mean = 0, sd = 10)
x2 <- 0.7 * xi + rnorm(n = n, mean = 0, sd = 10)
x3 <- 0.9 * xi + rnorm(n = n, mean = 0, sd = 10)
y1 <- 0.7 * eta + rnorm(n = n, mean = 0, sd = 10)
```

```
y2 <- 0.7 * eta + rnorm(n = n, mean = 0, sd = 10)
y3 <- 0.8 * eta + rnorm(n = n, mean = 0, sd = 10)

dfobs <- data.frame(x1, x2, x3, y1, y2, y3)

mec <- '
# Measurement models
  xi1  =~ 1*x1 + l2*x2 + l3*x3
  eta1 =~ 1*y1 + l2*y2 + l3*y3
# Regression at latent variable-level
  eta1 ~ gamma*xi1
# Variances
  xi1  ~~ phi*xi1
  eta1 ~~ psi*eta1
# Measurement error
  x1 ~~ d1*x1
  x2 ~~ d2*x2
  x3 ~~ d3*x3
  y1 ~~ e1*y1
  y2 ~~ e2*y2
  y3 ~~ e3*y3
'
mec.fit <- sem(model = mec, data = dfobs)
summary(mec.fit)
```

```
...
## Regressions:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   eta1 ~
##     xi1     (gamm)    0.034    0.731    0.047    0.963
...
```

Notation and path diagrams

There exists a formalized structure for constructing the model equations in the form of a *structural model*

$$\boldsymbol{\eta} = \boldsymbol{B}\boldsymbol{\eta} + \boldsymbol{\Gamma}\boldsymbol{\xi} + \boldsymbol{\zeta}$$

where $\boldsymbol{\eta}$ is a vector of all the dependent or *endogenous* latent variables and $\boldsymbol{\xi}$ is a vector of the independent or *exogenous* latent variables, $\boldsymbol{B}$ is a matrix of regression coefficients linking endogenous latent variables (think of a mediation model with a latent mediator) and $\boldsymbol{\Gamma}$ are the regression coefficients linking the endogenous to the exogenous latent variables. Finally, $\boldsymbol{\zeta}$ are the errors or *disturbances* of the endogenous latent variables.

Now, the latent variables are not observed, so we have to infer them from their observed measures. This is described in the *measurement models*

$$\begin{aligned}\boldsymbol{y} &= \boldsymbol{\Lambda}_y\boldsymbol{\eta} + \boldsymbol{\epsilon},\\ \boldsymbol{x} &= \boldsymbol{\Lambda}_x\boldsymbol{\xi} + \boldsymbol{\delta}\end{aligned}$$

where $\boldsymbol{y}$ and $\boldsymbol{x}$ are vectors of the observed dependent and independent variables, respectively. These are linked to the latent variables by *factor loadings* contained in the matrices $\boldsymbol{\Lambda}_y$ and $\boldsymbol{\Lambda}_x$. The *measurement error*, i.e., the part of the observed variables not explained by the latent variables, is $\boldsymbol{\epsilon}$ and $\boldsymbol{\delta}$.

Take a simple model with two latent variables, one exogenous $\boldsymbol{\xi} = (\xi)$, one endogenous $\boldsymbol{\eta} = (\eta)$, each measured by three observed variables, $\boldsymbol{x} = (x_1, x_2, x_3)^\intercal$, $\boldsymbol{y} = (y_1, y_2, y_3)^\intercal$. Then we would write the structural model as

$$\begin{aligned}\boldsymbol{\eta} &= \boldsymbol{B}\boldsymbol{\eta} + \boldsymbol{\Gamma}\boldsymbol{\xi} + \boldsymbol{\zeta} \\ \eta &= \gamma\xi + \zeta\end{aligned}$$

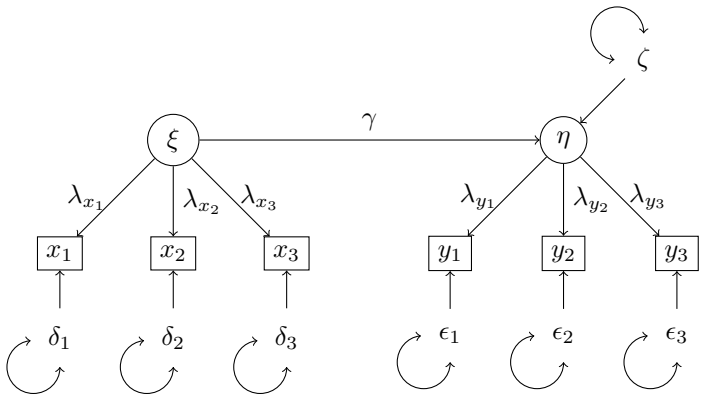
and the measurement models as

$$\begin{aligned}\boldsymbol{y} &= \boldsymbol{\Lambda}_y\boldsymbol{\eta} + \boldsymbol{\epsilon} \\ \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} &= \begin{bmatrix} \lambda_{y_1} \\ \lambda_{y_2} \\ \lambda_{y_3} \end{bmatrix} \eta + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \end{bmatrix}, \\ \boldsymbol{x} &= \boldsymbol{\Lambda}_x\boldsymbol{\xi} + \boldsymbol{\delta} \\ \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} &= \begin{bmatrix} \lambda_{x_1} \\ \lambda_{x_2} \\ \lambda_{x_3} \end{bmatrix} \xi + \begin{bmatrix} \delta_1 \\ \delta_2 \\ \delta_3 \end{bmatrix}.\end{aligned}$$

We would typically state further our assumptions about the models, i.e., that the errors and disturbances are mean independent or at least uncorrelated with the predictors, i.e., $\mathbb{E}(\zeta|\xi) = \mathbb{E}(\zeta)$, or $\text{Cov}(\zeta, \xi) = 0$, and that the unexplained parts of the observed variables, the measurement error, are mutually unrelated, e.g., $\text{Cov}(\epsilon_1, \epsilon_2) = \text{Cov}(\epsilon_1, \epsilon_3) = \text{Cov}(\epsilon_2, \epsilon_3) = \text{Cov}(\delta_1, \delta_2) = \text{Cov}(\delta_1, \delta_3) = \text{Cov}(\delta_2, \delta_3) = 0$.

We can again use *path diagrams* to succinctly describe the relations and assumptions of our model.

Figure 1: Example path diagram



Model-implied covariance matrix

Once we have specified the model using the typical SEM notation, we can derive the model-implied covariance matrix. This means working with matrices and vectors, so the algebra may be difficult for some to follow. Essentially, with mean-centered variables, we can say the covariance matrix of the observed endogenous variables, call it $\boldsymbol{\Sigma}_{yy}(\boldsymbol{\theta})$, is $\mathbb{E}(\boldsymbol{y}\boldsymbol{y}^\intercal)$ where $^\intercal$ is the transpose operator, changing, say, a $p \times 1$ vector into a $1 \times p$ vector.[1] So, we express the variance of $\boldsymbol{y}$ in terms of the model,

$$\begin{aligned}\boldsymbol{\Sigma}_{yy}(\boldsymbol{\theta}) &= \mathbb{E}(\boldsymbol{y}\boldsymbol{y}^\intercal) \\ &= \mathbb{E}[(\boldsymbol{\Lambda}_y\boldsymbol{\eta} + \boldsymbol{\epsilon})(\boldsymbol{\Lambda}_y\boldsymbol{\eta} + \boldsymbol{\epsilon})^\intercal] \\ &= \mathbb{E}[(\boldsymbol{\Lambda}_y\boldsymbol{\eta} + \boldsymbol{\epsilon})(\boldsymbol{\eta}^\intercal\boldsymbol{\Lambda}_y^\intercal + \boldsymbol{\epsilon}^\intercal)] \\ &= \mathbb{E}(\boldsymbol{\Lambda}_y\boldsymbol{\eta}\boldsymbol{\eta}^\intercal\boldsymbol{\Lambda}_y^\intercal + \boldsymbol{\Lambda}_y\boldsymbol{\eta}\boldsymbol{\epsilon}^\intercal + \boldsymbol{\epsilon}\boldsymbol{\Lambda}_y^\intercal\boldsymbol{\eta}^\intercal + \boldsymbol{\epsilon}\boldsymbol{\epsilon}^\intercal)\end{aligned}$$

[1] An imprtant property of transposing is if we have $(\boldsymbol{u}\boldsymbol{v})$, then the transpose is $(\boldsymbol{u}\boldsymbol{v})^\intercal = (\boldsymbol{v}^\intercal\boldsymbol{u}^\intercal)$.

where $\mathbb{E}(\boldsymbol{\eta}\boldsymbol{\epsilon}) = \text{Cov}(\boldsymbol{\eta}, \boldsymbol{\epsilon}) = \mathbf{0}$, by assumption, so

$$\begin{aligned}
\boldsymbol{\Sigma}_{yy}(\boldsymbol{\theta}) &= \mathbb{E}(\boldsymbol{\Lambda}_y \boldsymbol{\eta}\boldsymbol{\eta}^\intercal \boldsymbol{\Lambda}_y^\intercal + \boldsymbol{\epsilon}\boldsymbol{\epsilon}^\intercal) \\
&= \boldsymbol{\Lambda}_y \, \mathbb{E}(\boldsymbol{\eta}\boldsymbol{\eta}^\intercal) \boldsymbol{\Lambda}_y^\intercal + \mathbb{E}(\boldsymbol{\epsilon}\boldsymbol{\epsilon}^\intercal)
\end{aligned}$$

We call $\mathbb{E}(\boldsymbol{\epsilon}\boldsymbol{\epsilon}^\intercal) = \boldsymbol{\Theta}_\epsilon$, the covariance matrix of the the measurement errors for $\boldsymbol{y}$. As per our assumption (noted above and reflected in the path diagram), this matrix will have the variances of the measurement errors on the diagonal and zeros everywhere else, implying the measurement errors are uncorrelated with each other (the error of one indicator does not tell us anything about the error of another indicator).

Further, we can expand on $\mathbb{E}(\boldsymbol{\eta}\boldsymbol{\eta}^\intercal)$ by putting $\boldsymbol{\eta}$ in reduced form

$$\begin{aligned}
\boldsymbol{\eta} &= \boldsymbol{B}\boldsymbol{\eta} + \boldsymbol{\Gamma}\boldsymbol{\xi} + \boldsymbol{\zeta} \\
\boldsymbol{\eta} - \boldsymbol{B}\boldsymbol{\eta} &= \boldsymbol{\Gamma}\boldsymbol{\xi} + \boldsymbol{\zeta} \\
\boldsymbol{\eta}(\boldsymbol{I} - \boldsymbol{B}) &= \boldsymbol{\Gamma}\boldsymbol{\xi} + \boldsymbol{\zeta} \\
(\boldsymbol{I} - \boldsymbol{B})^{-1}\boldsymbol{\eta}(\boldsymbol{I} - \boldsymbol{B}) &= (\boldsymbol{I} - \boldsymbol{B})^{-1}(\boldsymbol{\Gamma}\boldsymbol{\xi} + \boldsymbol{\zeta}) \\
\boldsymbol{\eta} &= (\boldsymbol{I} - \boldsymbol{B})^{-1}(\boldsymbol{\Gamma}\boldsymbol{\xi} + \boldsymbol{\zeta})
\end{aligned}$$

which we substitute back into $\mathbb{E}(\boldsymbol{\eta}\boldsymbol{\eta}^\intercal)$

$$\begin{aligned}
\boldsymbol{\Sigma}_{yy}(\boldsymbol{\theta}) &= \boldsymbol{\Lambda}_y \, \mathbb{E}(\boldsymbol{\eta}\boldsymbol{\eta}^\intercal) \boldsymbol{\Lambda}_y^\intercal + \boldsymbol{\Theta}_\epsilon \\
&= \boldsymbol{\Lambda}_y \, \mathbb{E}[((\boldsymbol{I} - \boldsymbol{B})^{-1}(\boldsymbol{\Gamma}\boldsymbol{\xi} + \boldsymbol{\zeta}))((\boldsymbol{I} - \boldsymbol{B})^{-1}(\boldsymbol{\Gamma}\boldsymbol{\xi} + \boldsymbol{\zeta}))^\intercal)] \boldsymbol{\Lambda}_y^\intercal + \boldsymbol{\Theta}_\epsilon \\
&= \boldsymbol{\Lambda}_y \, \mathbb{E}[((\boldsymbol{I} - \boldsymbol{B})^{-1}\boldsymbol{\Gamma}\boldsymbol{\xi} + (\boldsymbol{I} - \boldsymbol{B})^{-1}\boldsymbol{\zeta})(\boldsymbol{\xi}^\intercal \boldsymbol{\Gamma}^\intercal (\boldsymbol{I} - \boldsymbol{B})^{-1\intercal} + \boldsymbol{\zeta}^\intercal (\boldsymbol{I} - \boldsymbol{B})^{-1\intercal})] + \boldsymbol{\Lambda}_y^\intercal + \boldsymbol{\Theta}_\epsilon \\
&= \boldsymbol{\Lambda}_y \, \mathbb{E}[(\boldsymbol{I} - \boldsymbol{B})^{-1}(\boldsymbol{\Gamma}\boldsymbol{\xi}\boldsymbol{\xi}^\intercal \boldsymbol{\Gamma}^\intercal + \boldsymbol{\zeta}\boldsymbol{\zeta}^\intercal)(\boldsymbol{I} - \boldsymbol{B})^{-1\intercal}] \boldsymbol{\Lambda}_y^\intercal + \boldsymbol{\Theta}_\epsilon \\
&= \boldsymbol{\Lambda}_y (\boldsymbol{I} - \boldsymbol{B})^{-1}(\boldsymbol{\Gamma} \, \mathbb{E}(\boldsymbol{\xi}\boldsymbol{\xi}^\intercal) \boldsymbol{\Gamma}^\intercal + \mathbb{E}(\boldsymbol{\zeta}\boldsymbol{\zeta}^\intercal))(\boldsymbol{I} - \boldsymbol{B})^{-1\intercal} \boldsymbol{\Lambda}_y^\intercal + \boldsymbol{\Theta}_\epsilon \\
&= \boldsymbol{\Lambda}_y (\boldsymbol{I} - \boldsymbol{B})^{-1}(\boldsymbol{\Gamma}\boldsymbol{\Phi}\boldsymbol{\Gamma}^\intercal + \boldsymbol{\Psi})(\boldsymbol{I} - \boldsymbol{B})^{-1\intercal} \boldsymbol{\Lambda}_y^\intercal + \boldsymbol{\Theta}_\epsilon
\end{aligned}$$

where $\boldsymbol{\Phi}$ is the covariance matrix of the latent exogenous variables and $\boldsymbol{\Psi}$ is again the covariance matrix of the disturbances. This is the model-implied covariance matrix for the vector of endogenous observed variables. For the exogenous observed variables, we can proceed in a similar fashion and obtain

$$\begin{aligned}
\boldsymbol{\Sigma}_{xx}(\boldsymbol{\theta}) &= \mathbb{E}(\boldsymbol{x}\boldsymbol{x}^\intercal) \\
&= \mathbb{E}[(\boldsymbol{\Lambda}_x\boldsymbol{\xi} + \boldsymbol{\delta})(\boldsymbol{\Lambda}_x\boldsymbol{\xi} + \boldsymbol{\delta})^\intercal] \\
&= \mathbb{E}[(\boldsymbol{\Lambda}_x\boldsymbol{\xi} + \boldsymbol{\delta})(\boldsymbol{\xi}^\intercal\boldsymbol{\Lambda}_x^\intercal + \boldsymbol{\delta}^\intercal)] \\
&= \boldsymbol{\Lambda}_x\boldsymbol{\Phi}\boldsymbol{\Lambda}_x^\intercal + \boldsymbol{\Theta}_\delta
\end{aligned}$$

(since $\mathbb{E}(\boldsymbol{\xi}\boldsymbol{\delta}^\intercal) = \mathbb{E}(\boldsymbol{\delta}\boldsymbol{\xi}^\intercal) = \mathbf{0}$) where, again, $\boldsymbol{\Phi}$ is the covariance matrix of the latent exogenous variables and $\boldsymbol{\Theta}_\delta$ is the covariance matrix of the measurement errors for x. Finally, we have

$$\begin{aligned}
\boldsymbol{\Sigma}_{yx}(\boldsymbol{\theta}) &= \mathbb{E}(\boldsymbol{y}\boldsymbol{x}^\intercal) \\
&= \mathbb{E}[(\boldsymbol{\Lambda}_y\boldsymbol{\eta} + \boldsymbol{\epsilon})(\boldsymbol{\Lambda}_x\boldsymbol{\xi} + \boldsymbol{\delta})^\intercal] \\
&= \mathbb{E}[(\boldsymbol{\Lambda}_y\boldsymbol{\eta} + \boldsymbol{\epsilon})(\boldsymbol{\xi}^\intercal\boldsymbol{\Lambda}^\intercal + \boldsymbol{\delta}^\intercal)] \\
&= \boldsymbol{\Lambda}_y \, \mathbb{E}(\boldsymbol{\eta}\boldsymbol{\xi}^\intercal)\boldsymbol{\Lambda}_x^\intercal.
\end{aligned}$$

Remember, though, that $\boldsymbol{\eta} = (\boldsymbol{I} - \boldsymbol{B})^{-1}(\boldsymbol{\Gamma}\boldsymbol{\xi} + \boldsymbol{\zeta})$, so

$$\begin{aligned}
\mathbb{E}(\boldsymbol{\eta}\boldsymbol{\xi}^\intercal) &= \mathbb{E}[((\boldsymbol{I} - \boldsymbol{B})^{-1}(\boldsymbol{\Gamma}\boldsymbol{\xi} + \boldsymbol{\zeta}))\boldsymbol{\xi}^\intercal] \\
&= (\boldsymbol{I} - \boldsymbol{B})^{-1}\boldsymbol{\Gamma} \, \mathbb{E}(\boldsymbol{\xi}\boldsymbol{\xi}^\intercal) \\
&= (\boldsymbol{I} - \boldsymbol{B})^{-1}\boldsymbol{\Gamma}\boldsymbol{\Phi}
\end{aligned}$$

(since $\mathbb{E}(\boldsymbol{\zeta}\boldsymbol{\xi}^\intercal) = \mathbf{0}$) so we have

$$\boldsymbol{\Sigma}_{yx}(\boldsymbol{\theta}) = \boldsymbol{\Lambda}_y(\boldsymbol{I} - \boldsymbol{B})^{-1}\boldsymbol{\Gamma}\boldsymbol{\Phi}\boldsymbol{\Lambda}_x^\intercal.$$

Now, we put these all together into the model-implied covariance matrix, which is subdivided into the matrices we just worked out

$$\begin{aligned}\boldsymbol{\Sigma}(\boldsymbol{\theta}) &= \begin{bmatrix}\boldsymbol{\Sigma}_{yy}(\boldsymbol{\theta}) & \boldsymbol{\Sigma}_{yx}(\boldsymbol{\theta}) \\ \boldsymbol{\Sigma}_{xy}(\boldsymbol{\theta}) & \boldsymbol{\Sigma}_{xx}(\boldsymbol{\theta})\end{bmatrix} \\ &= \begin{bmatrix}\boldsymbol{\Lambda}_y(\boldsymbol{I} - \boldsymbol{B})^{-1}(\boldsymbol{\Gamma}\boldsymbol{\Phi}\boldsymbol{\Gamma}^\intercal + \boldsymbol{\Psi})(\boldsymbol{I} - \boldsymbol{B})^{-1\intercal}\boldsymbol{\Lambda}_y^\intercal + \boldsymbol{\Theta}_\epsilon & \boldsymbol{\Lambda}_y(\boldsymbol{I} - \boldsymbol{B})^{-1}\boldsymbol{\Gamma}\boldsymbol{\Phi}\boldsymbol{\Lambda}_x^\intercal \\ \boldsymbol{\Lambda}_x\boldsymbol{\Phi}\boldsymbol{\Gamma}^\intercal(\boldsymbol{I} - \boldsymbol{B})^{-1\intercal}\boldsymbol{\Lambda}_y^\intercal & \boldsymbol{\Lambda}_x\boldsymbol{\Phi}\boldsymbol{\Lambda}_x^\intercal + \boldsymbol{\Theta}_\delta\end{bmatrix}.\end{aligned}$$

Notice one of the reasons why it is called the 'general' SEM: if we assume the variables are measured without error, i.e., $\boldsymbol{\Theta}_\epsilon = \boldsymbol{\Theta}_\delta = \mathbf{0}$ and $\boldsymbol{\Lambda}_y = \boldsymbol{I}$ and $\boldsymbol{\Lambda}_x = \boldsymbol{I}$, then the equations reduce to the linear and mediation models with observed variables we started the course with. If we set $\boldsymbol{B} = \boldsymbol{\Gamma} = \mathbf{0}$, $\boldsymbol{\Theta}_\epsilon = \mathbf{0}$, $\boldsymbol{\Lambda}_y = \mathbf{0}$ and $\boldsymbol{\Psi} = \mathbf{0}$ then we get the confirmatory factor analysis model.

Identification

As the models become more complicated, the issue of identification becomes even more important. Take the hypothetical example shown in Figure 1.

Remember, regardless of how many latent variables and errors are shown in the model, the only pieces of empirical information are the observed variances and covariances. In this example, we have three exogenous indicators and three endogenous indicators. By looking at a covariance matrix or by using the formula $p(p+1)/2$, where p is the number of observed variables overall, we have $6(6+1)/2 = 6(7)/2 = 42/2 = 21$ pieces of empirical information. We can estimate up to 21 parameters for a just-identified model.

It is easiest to refer to the path diagram to find the number of parameters to estimate. We have six measurement error variances, four factor loadings (since we will be fixing one factor loading on each measurement model to 1.0), one regression coefficient, the variance of the exogenous latent variable and the disturbance variance. This gives $6 + 4 + 1 + 1 + 1 = 13$ parameters to be estimated, so we have 21 - 13 = 8 degrees of freedom. The model is thus over-identified and we can assess its fit.

Empirical example

Most sociological, as well as psychological concepts cannot be measured directly. We may try to measure such constructs using, say, questions in a survey, but for the most part, we must admit that our measures will tend to be crude approximations of the underlying concept and that we typically have to deal with measurement error.

Researchers have known this for many years though, and their is the entire field of psychometrics whose work looks at and assesses measurements of more or less abstract concepts. This is why most social surveys assess concepts like xenophobia, need for social approval, environmental attitudes, etc. using scales whose properties (reliability and validity) have been tested to certain extents. GESIS, for example, hosts an open access repository for measurement instruments called the Zusammenstellung sozialwissenschaftlicher Items und Skalen (ZIS, https://zis.gesis.org/). ZIS collects indicators meant to measure a specific concept whose properties have been tested systematically in the past.

We now expand our measures of xenophobia and populism to multiple indicator measurement models and specify the general SEM. We will bring social class back in as an observed predictor in the next section. To measure xenophobia, we will use:

- `px06`: "There is a dangerous amount of foreigners living in Germany"

Table 1: Notation

Symbol	Pronunciation	Explanation
$\boldsymbol{x}$		vector of exogenous observed variables
$\boldsymbol{y}$		vector of endogenous observed variables
$\boldsymbol{\xi}$	ksi	vector of exogenous latent variables
$\boldsymbol{\eta}$	eta	vector of endogenous latent variables
$\boldsymbol{\zeta}$	zeta	vector of disturbances
$\boldsymbol{\delta}$	delta	vector of exogenous measurement errors
$\boldsymbol{\epsilon}$	epsilon	vector of endogenous measurement errors
x		exogenous (independent) observed variable
y		endogenous (dependent) observed variable
ξ	ksi	exogenous latent variable
η	eta	endogenous latent variable
ζ	zeta	error or disturbance of endogenous latent variable
δ	delta	error of exogenous observed variable
ϵ	epsilon	error of endogenous observed latent variable
$\boldsymbol{\Lambda}_x$	Lambda x	factor loading matrix for exogenous variables
$\boldsymbol{\Lambda}_y$	Lambda y	factor loading matrix for endogenous variables
$\boldsymbol{\Phi}$	Phi	covariance matrix of exogenous latent variables
$\boldsymbol{\Psi}$	Psi	covariance matrix of endogenous variables (errors/disturbances)
$\boldsymbol{\Theta}_\delta$	Theta delta	covariance matrix of the exogenous errors
$\boldsymbol{\Theta}_\epsilon$	Theta epsilon	covariance matrix of the endogenous variables (errors)
τ_x	tau x	intercept of observed exogenous variable
τ_y	tau y	intercept of observed endogenous variable
λ_x	lambda x	element of matrix $\boldsymbol{\Lambda}_x$
λ_y	lambda y	element of matrix $\boldsymbol{\Lambda}_y$
θ_δ	theta delta	element of matrix $\boldsymbol{\Theta}_\delta$
θ_ϵ	theta delta	element of matrix $\boldsymbol{\Theta}_\epsilon$
ϕ	phi	element of matrix $\boldsymbol{\Phi}$
ψ	psi	element of matrix $\boldsymbol{\Psi}$

- `px07`: "Foreigners should marry amongst their own people"
- `px10`: "Attacks on asylum-seeker housing are understandable"

where 1: "fully disagree," ..., 5: "fully agree." For populist sentiment, we will use the following indicators:

- `pa30r`: "Politicians talk too much instead of acting (recoded)"
- `pa32r`: "Political compromise is a betrayal of principals (recoded)"
- `pa35r`: "Politicians only represent the rich (recoded)"

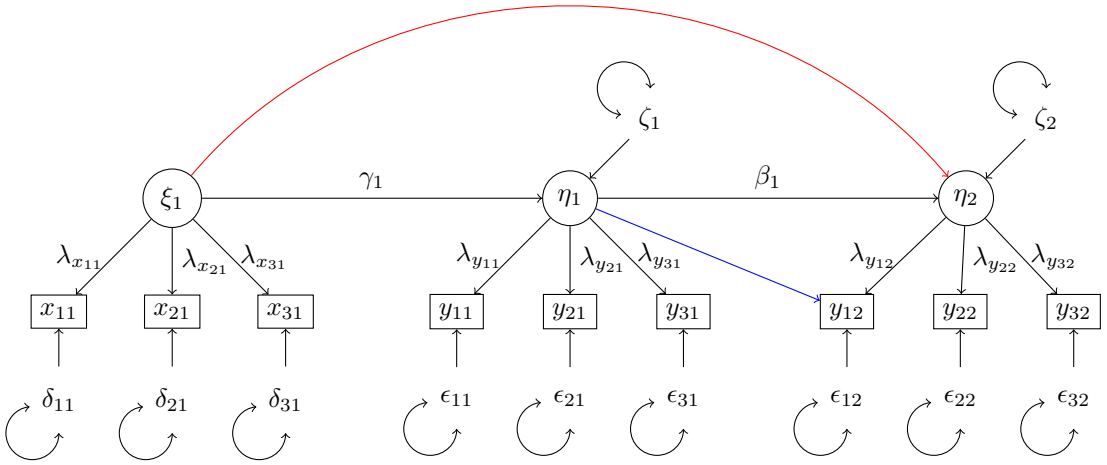
where also 1: "fully disagree," ..., 5: "fully agree."

The procedure for testing an SEM normally involves *two steps*.

1. Fit a CFA of all the latent constructs of the model.
2. If the fit of the CFA is adequate proceed with the SEM.

We usually proceed in two steps because once we have fit the SEM, misfit could be caused by either a) the measurement models or b) the structural relations. Take the path diagram in Figure 2 as an example:

Figure 2: Misfit due to measurement (blue) or structural (red) portion



If we fit the entire model simultaneously and find that the fit is inadequate, it will be more difficult for us to determine whether the misfit is due to a problem in the measurement model, or whether it is due to some structural parameter. For example, maybe an indicator loads strongly on two different latent variables (cross-loading). This is shown in Figure 2 with the blue factor loading going from η_1 to y_{12}. Fixing the cross-loading to zero (since we normally assume each indicator loads on one and only one latent variable) will be a source of misfit. The structural part might be correctly specified but our model will fit suboptimally.

On the other hand, our measurement model might be correctly specified but our structural model may be misspecified. In Figure 2, we are (implicitly) assuming η_1 is a total mediator of the effect of ξ_1 on η_2. If this assumption is wrong, and there is an effect over and above the mediator (the direct effect of ξ_1 on η_2, shown in red), then our structural model will be misspecified and our model will fit suboptimally.

The solution to this problem is to first *fit a CFA with all the latent variables simultaneously* (notice this is not the same as estimating a separate CFA for each construct). We *allow the latent variables to covary with each other freely.* Allowing them to all covary with each other means the structural portion of the model is saturated; if we allow ξ_1 to covary with η_1 and η_2 and η_1 to covary with η_2 then we have zero degrees of freedom and the *fit at the structural level will be perfect.* If the model nevertheless does not fit adequately, then we know that the source of misfit is at the measurement level — it is the only level on which we are imposing any constraints.

Simultaneous CFA

Let us now fit the simultaneous CFA for populism and xenophobia.

```
# Load packages
library(haven)
library(lavaan)

# Set working directory
setwd("../04_data")

# Import ALLBUS 2018
df <- read_sav("allbus2018.sav")

# Simultaneous CFA
cfa1 <- '
# Measurement models
  populism   =~ 1*pa30r + l21*pa32r + l31*pa35r
  xenophobia =~ 1*px06  + l22*px07  + l32*px10
# Exogenous variances
  populism   ~~ phi11*populism
  xenophobia ~~ phi22*xenophobia
# Error variances
  pa30r ~~ theta11*pa30r
  pa32r ~~ theta21*pa32r
  pa35r ~~ theta31*pa35r
  px06  ~~ theta12*px06
  px07  ~~ theta22*px07
  px10  ~~ theta32*px10
# Covariance
  populism ~~ phi21*xenophobia
'
cfa1.fit <- cfa(model = cfa1, data = df, estimator = "ML")
```

Remember the syntax here is much more complicated than it needs to be. The model could be estimated with as little as

```
cfa1 <- '
populism =~ pa30r + pa32r + pa35r
xenophobia =~ px06 + px07  + px10
populism ~~ xenophobia
'
```

Now let us look at the results.

```
summary(cfa1.fit, fit.measures = TRUE, standardized = TRUE)
```

```
## lavaan 0.6-8 ended normally after 29 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                        13
##
```

```
##                                                   Used       Total
##   Number of observations                          3072        3477
##
## Model Test User Model:
##
##   Test statistic                                91.128
##   Degrees of freedom                                 8
##   P-value (Chi-square)                           0.000
##
## Model Test Baseline Model:
##
##   Test statistic                              4207.334
##   Degrees of freedom                                15
##   P-value                                        0.000
##
## User Model versus Baseline Model:
##
##   Comparative Fit Index (CFI)                    0.980
##   Tucker-Lewis Index (TLI)                       0.963
##
## Loglikelihood and Information Criteria:
##
##   Loglikelihood user model (H0)             -26059.125
##   Loglikelihood unrestricted model (H1)             NA
##
##   Akaike (AIC)                               52144.251
##   Bayesian (BIC)                             52222.642
##   Sample-size adjusted Bayesian (BIC)        52181.335
##
## Root Mean Square Error of Approximation:
##
##   RMSEA                                          0.058
##   90 Percent confidence interval - lower         0.048
##   90 Percent confidence interval - upper         0.069
##   P-value RMSEA <= 0.05                          0.096
##
## Standardized Root Mean Square Residual:
##
##   SRMR                                           0.026
##
## Parameter Estimates:
##
##   Standard errors                             Standard
##   Information                                 Expected
##   Information saturated (h1) model          Structured
##
## Latent Variables:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##   populism =~
##     pa30r             1.000                               0.697    0.703
##     pa32r    (l21)    1.209    0.039   30.616    0.000    0.842    0.734
##     pa35r    (l31)    1.115    0.037   30.106    0.000    0.777    0.701
##   xenophobia =~
##     px06              1.000                               1.071    0.757
```

```
##     px07     (l22)    0.679    0.029   23.245    0.000    0.727    0.625
##     px10     (l32)    0.378    0.020   18.955    0.000    0.405    0.443
##
## Covariances:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##   populism ~~
##     xenophb (ph21)    0.466    0.024   19.798    0.000    0.624    0.624
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##     populsm (ph11)    0.485    0.025   19.288    0.000    1.000    1.000
##     xenophb (ph22)    1.148    0.063   18.246    0.000    1.000    1.000
##    .pa30r   (th11)    0.495    0.018   27.420    0.000    0.495    0.505
##    .pa32r   (th21)    0.607    0.024   25.138    0.000    0.607    0.461
##    .pa35r   (th31)    0.623    0.023   27.557    0.000    0.623    0.508
##    .px06    (th12)    0.853    0.048   17.803    0.000    0.853    0.426
##    .px07    (th22)    0.826    0.029   28.058    0.000    0.826    0.610
##    .px10    (th32)    0.673    0.019   35.389    0.000    0.673    0.804
```

From this, we see that the measurement model for populism works quite well. If we take the more conservative rule of thumb that each standardized factor loading should be > 0.7, the model is adequate. The model for xenophobia, on the other hand, is less than optimal. The standardized factor loading for `px10` is < 0.5, which indicates the inter-item correlation between it and the other items is potentially low.

Looking at the fit indices, we see the chi square test is 91.128 with 8 degrees of freedom and significant. This suggests we should reject the null hypothesis that the model-implied covariance matrix matches the observed matrix. On the other hand, the CFI and TLI measures are larger than 0.950, the RMSEA is only just over 0.05, and the SRMR is well below 0.05. Taken together, we could conclude that the fit is suboptimal but not terrible. We may want to proceed with the model nevertheless, especially if we consider the variable `px10` as an essential indicator for xenophobia from a theoretical standpoint.

In other words, sometimes there are theoretical arguments for including indicators. For example, the classic definition of authoritarianism involves a subservience to those perceived above one's own standing and a desire to dominate those perceived to be below. In such a case, we would have a theoretical argument for including indicators of both characteristics, which may trump issues of fit.

Here, there is probably no good theoretical argument for why an acceptance of attacks on asylum-seeker housing is an essential indicator of xenophobia. We will keep the indicator in for now, knowing full well that the measurement model could be improved by either dropping the indicator or replacing it with another one.

Full SEM

Assuming we were satisfied with the fit of the CFA, we could move on to the full SEM. We simply need to use what we have learned up until now at the latent variable-level. When we want to regress a latent variable on another, we simply use the `~` regression operator, just as before. The only difference between the CFA and the SEM in this case is that the correlation will now be a regression coefficient.[2]

```
# Full SEM
sem1 <- '
# Measurement models
  populism   =~ 1*pa30r + lx21*pa32r + lx31*pa35r
```

[2]Notice that in this rather simple example, the procedure of first estimating the CFA and then the SEM is redundant! Whether we allow the latent variables to covary or regress one on the other, the covariance between the two will be captured. The two-step procedure only really makes sense when some constraints are being applied to the structural level.

```
  xenophobia =~ 1*px06  + ly21*px07  + ly31*px10
# Regression
  xenophobia ~ gamma1*populism
# Exogenous variances
  populism   ~~ phi11*populism
# Disturbance variance
  xenophobia ~~ psi11*xenophobia
# Error variances
  pa30r ~~ theta11*pa30r
  pa32r ~~ theta21*pa32r
  pa35r ~~ theta31*pa35r
  px06  ~~ theta12*px06
  px07  ~~ theta22*px07
  px10  ~~ theta32*px10
'
sem1.fit <- sem(model = sem1, data = df, estimator = "ML")
summary(sem1.fit, fit.measures = TRUE, standardized = TRUE)
```

```
## lavaan 0.6-8 ended normally after 26 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                        13
##
##                                                   Used       Total
##   Number of observations                          3072        3477
##
## Model Test User Model:
##
##   Test statistic                                91.128
##   Degrees of freedom                                 8
##   P-value (Chi-square)                           0.000
##
## Model Test Baseline Model:
##
##   Test statistic                              4207.334
##   Degrees of freedom                                15
##   P-value                                        0.000
##
## User Model versus Baseline Model:
##
##   Comparative Fit Index (CFI)                    0.980
##   Tucker-Lewis Index (TLI)                       0.963
##
## Loglikelihood and Information Criteria:
##
##   Loglikelihood user model (H0)             -26059.125
##   Loglikelihood unrestricted model (H1)             NA
##
##   Akaike (AIC)                               52144.251
##   Bayesian (BIC)                             52222.642
##   Sample-size adjusted Bayesian (BIC)        52181.335
##
```

```
## Root Mean Square Error of Approximation:
##
##   RMSEA                                          0.058
##   90 Percent confidence interval - lower         0.048
##   90 Percent confidence interval - upper         0.069
##   P-value RMSEA <= 0.05                          0.096
##
## Standardized Root Mean Square Residual:
##
##   SRMR                                           0.026
##
## Parameter Estimates:
##
##   Standard errors                             Standard
##   Information                                 Expected
##   Information saturated (h1) model          Structured
##
## Latent Variables:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##   populism =~
##     pa30r             1.000                               0.697    0.703
##     pa32r   (lx21)    1.209    0.039   30.616    0.000    0.842    0.734
##     pa35r   (lx31)    1.115    0.037   30.106    0.000    0.777    0.701
##   xenophobia =~
##     px06              1.000                               1.071    0.757
##     px07    (ly21)    0.679    0.029   23.245    0.000    0.727    0.625
##     px10    (ly31)    0.378    0.020   18.955    0.000    0.405    0.443
##
## Regressions:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##   xenophobia ~
##     populsm (gmm1)    0.960    0.043   22.412    0.000    0.624    0.624
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##     populsm (ph11)    0.485    0.025   19.288    0.000    1.000    1.000
##    .xenophb (ps11)    0.700    0.048   14.520    0.000    0.610    0.610
##    .pa30r   (th11)    0.495    0.018   27.420    0.000    0.495    0.505
##    .pa32r   (th21)    0.607    0.024   25.138    0.000    0.607    0.461
##    .pa35r   (th31)    0.623    0.023   27.557    0.000    0.623    0.508
##    .px06    (th12)    0.853    0.048   17.803    0.000    0.853    0.426
##    .px07    (th22)    0.826    0.029   28.058    0.000    0.826    0.610
##    .px10    (th32)    0.673    0.019   35.389    0.000    0.673    0.804
```

Again, it is worth highlighting the fact that this model is essentially identical to the simultaneous CFA from above since all we have done is change a correlation to a regression effect. The fit, measurement models and error variances are all identical except now instead of estimating the variance of xenophobia, we are estimating the disturbance variance, the part that is unrelated to populism.

Nevertheless, we can see a strong positive effect of populism on xenophobia, with an unstandardized effect of 0.960^{***} (0.043) that translates to a standardized effect of 0.624. An increase of one standard deviation in populism leads to a 0.624 standard deviation increase in xenophobia.

Since the scales of the latent variables are rather arbitrary, we tend to focus on the standardized effects and interpret the magnitude of the effect, first and foremost.

Including observed predictors: the MIMIC model

The Multiple Indicators Multiple Causes (MIMIC) model is an extension of the general SEM in which we have observed covariates predicting at least one endogenous latent variable. Say we were interested in whether males or females tended to differ in terms of their populist sentiment. Then we would regress our latent populism variable on the observed variable for the respondent's sex.

Let us say we were willing to assume social class was measured without error. Then we could include it in the model as an observed predictor for xenophobia and populism just as in the mediation model. The implementation is straightforward, we simply regress the latent variables, now both endogenous, on the observed variable `id02`. This would be one example of a MIMIC model.

```
# MIMIC model
sem2 <- '
# Measurement models
  populism   =~ 1*pa30r + lx21*pa32r + lx31*pa35r
  xenophobia =~ 1*px06  + ly21*px07  + ly31*px10
# Regressions
  xenophobia ~ gamma1*id02 + beta1*populism # Change coefficient names around
  populism   ~ gamma2*id02
# Exogenous variances
  id02       ~~ phi11*id02 # Class is only exogenous variable now
# Disturbance variance
  xenophobia ~~ psi11*xenophobia # Both xeno and pop are endogenous now
  populism   ~~ psi22*populism
# Error variances
  pa30r ~~ theta11*pa30r
  pa32r ~~ theta21*pa32r
  pa35r ~~ theta31*pa35r
  px06  ~~ theta12*px06
  px07  ~~ theta22*px07
  px10  ~~ theta32*px10
'
sem2.fit <- sem(model = sem2, data = df, estimator = "ML")
summary(sem2.fit, fit.measures = TRUE, standardized = TRUE)
```

```
## lavaan 0.6-8 ended normally after 27 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                        16
##
##                                                   Used       Total
##   Number of observations                          3037        3477
##
## Model Test User Model:
##
##   Test statistic                               101.722
##   Degrees of freedom                                12
##   P-value (Chi-square)                           0.000
##
## Model Test Baseline Model:
##
##   Test statistic                              4652.605
```

```
##   Degrees of freedom                                21
##   P-value                                        0.000
##
## User Model versus Baseline Model:
##
##   Comparative Fit Index (CFI)                    0.981
##   Tucker-Lewis Index (TLI)                       0.966
##
## Loglikelihood and Information Criteria:
##
##   Loglikelihood user model (H0)             -28632.842
##   Loglikelihood unrestricted model (H1)             NA
##
##   Akaike (AIC)                               57297.683
##   Bayesian (BIC)                             57393.981
##   Sample-size adjusted Bayesian (BIC)        57343.143
##
## Root Mean Square Error of Approximation:
##
##   RMSEA                                          0.050
##   90 Percent confidence interval - lower         0.041
##   90 Percent confidence interval - upper         0.059
##   P-value RMSEA <= 0.05                          0.508
##
## Standardized Root Mean Square Residual:
##
##   SRMR                                           0.024
##
## Parameter Estimates:
##
##   Standard errors                             Standard
##   Information                                 Expected
##   Information saturated (h1) model          Structured
##
## Latent Variables:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##   populism =~
##     pa30r             1.000                               0.692    0.699
##     pa32r    (lx21)   1.209    0.039   30.840    0.000    0.837    0.730
##     pa35r    (lx31)   1.132    0.037   30.410    0.000    0.783    0.708
##   xenophobia =~
##     px06              1.000                               1.058    0.748
##     px07     (ly21)   0.691    0.029   23.755    0.000    0.730    0.628
##     px10     (ly31)   0.386    0.020   19.136    0.000    0.408    0.445
##
## Regressions:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##   xenophobia ~
##     id02     (gmm1)  -0.216    0.035   -6.166    0.000   -0.205   -0.138
##     populsm (bet1)    0.866    0.045   19.180    0.000    0.566    0.566
##   populism ~
##     id02     (gmm2)  -0.423    0.022  -19.551    0.000   -0.612   -0.413
##
## Variances:
```

```
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##     id02    (ph11)    0.455    0.012   38.968    0.000    0.455    1.000
##    .xenophb (ps11)    0.666    0.046   14.571    0.000    0.596    0.596
##    .populsm (ps22)    0.398    0.021   18.761    0.000    0.830    0.830
##    .pa30r   (th11)    0.502    0.018   28.069    0.000    0.502    0.512
##    .pa32r   (th21)    0.613    0.024   25.873    0.000    0.613    0.467
##    .pa35r   (th31)    0.612    0.022   27.480    0.000    0.612    0.499
##    .px06    (th12)    0.879    0.046   18.996    0.000    0.879    0.440
##    .px07    (th22)    0.819    0.029   28.109    0.000    0.819    0.606
##    .px10    (th32)    0.674    0.019   35.198    0.000    0.674    0.802
```

Here, we see the model fit has changed only slightly, with a still significant chi square statistic, but good CFI and TLI, as well as RMSEA and SRMR values. The rest of the model (e.g., measurement portion) has also changed only slightly.

The regressions are the main focus, and we see a significant negative effect of class on xenophobia with -0.216^{***} (0.035) which translates to a standardized effect of -0.138. For every unit class increases, xenophobia decreases by 0.216 'units.' When class increases by one standard deviation, xenophobia decreases by 0.138 standard deviations.

Note that it may not make much sense to think of social class in standard deviations because it is measured on a Likert scale with only five categories. If we want, we can interpret the column `Std.lv`, in which only the latent variable is standardized. The effect is -0.205 which means that for every unit increase in class, xenophobia decreases by 0.205 standard deviations.

The effect of class on populism is also significant and negative with an effect of -0.423^{***} (0.022) and a standardized effect (`Std.all`) of -0.413. Here, again, the column `Std.lv` may make the most sense to interpret: for every unit increase in class, populism decreases by 0.612 standard deviations.

We can compare the results of the MIMIC model with the mediation model from before.

```
# Rename variables for easier comparison
df$xenophobia <- df$px06
df$populism <- df$pa30r

# Re-fit the mediation model with observed variables
mm1 <- '
# Regressions
  xenophobia ~ gamma1*id02 + beta1*populism
  populism   ~ gamma2*id02
# Exogenous variance
  id02 ~~ phi11*id02
# Endogenous variances
  xenophobia ~~ psi11*xenophobia
  populism   ~~ psi22*populism
'
mm1.fit <- sem(model = mm1, data = df, estimator = "ML")
summary(mm1.fit, standardized = TRUE) # Fit measures redundant
```

```
## lavaan 0.6-8 ended normally after 21 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                         6
##
```

```
##                                                   Used       Total
##   Number of observations                          3169        3477
##
## 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|)   Std.lv  Std.all
##   xenophobia ~
##     id02    (gmm1)   -0.398    0.035  -11.310    0.000   -0.398   -0.190
##     populsm (bet1)    0.462    0.024   19.237    0.000    0.462    0.322
##   populism ~
##     id02    (gmm2)   -0.387    0.025  -15.421    0.000   -0.387   -0.264
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##     id02    (ph11)    0.453    0.011   39.806    0.000    0.453    1.000
##    .xenophb (ps11)    1.657    0.042   39.806    0.000    1.657    0.828
##    .populsm (ps22)    0.906    0.023   39.806    0.000    0.906    0.930
```

Notice that in the mediation model, both xenophobia and populism are assumed to be measured without error. This lead to an estimated effect of 0.462^{***} (0.024) of populism on xenophobia (standardized: 0.322). When we correct for measurement error, and compare this to the effect we see in the MIMIC model, we see that there is attenuation bias happening. In the MIMIC model, the effect is much stronger, with 0.866^{***} (0.045) and a standardized effect of 0.566.

Modification indices

In the model `sem2.fit`, we saw conflicting fit information. The chi square test was significant, suggesting misfit, but the rest of the fit indices, notably CFI and SRMR were quite good. TLI and RMSEA were acceptable but not optimal.

One way to deal with this — arguably the appropriate way — would be to go back to the theory and double-check that our model is properly specified from a substantive standpoint. In other words, does the theory suggest a direct effect of class on xenophobia over and above the mediated effect of populism? If we had held this direct path to zero, and the fit was poor, we could consider freeing this path and re-examining the fit.

On the measurement side, we typically have less to go on from a theory standpoint. Normally we work with the scales and variables we have available, and it is not uncommon for tested scales to perform suboptimally on new data or in new contexts. But a CFA is *confirmatory* rather than exploratory, after all, and it is not uncommon to fine-tune measurement models based on empirical evidence after the base model has been specified. For example, the factor loading for the indicator `px10` is rather low. If there is no good theoretical argument for keeping the indicator in the measurement model, and if we have enough degrees of freedom to spare, we might consider deleting this indicator. Whether or not we are able to capture the underlying

latent variable using just two indicators is not a question that can be answered empirically, and we would arguably prefer a well-fitting measurement model with many indicators vs. a well-fitting measurement model with just a few.

Another much more debatable approach is to *allow the data to speak for themselves*. In other words, completely isolated from theoretical considerations, we could proceed in a *completely exploratory fashion*, by either

- *introducing new parameters* to the model to account for more sources of covariation, or by
- *removing redundant parameters* to increase the parsimony of the model and the chi square to degrees of freedom ratio.

Adding new parameters

The former are often called Lagrange Multipliers or Test Scores. They essentially tell us the change in model fit would result in relaxing the constraints on our model. Normally these constraints entail setting certain parameters to zero. For example, in formulating our measurement models, we implicitly constrain all the possible cross-loadings (which are a feature of exploratory, rather than confirmatory factor analysis) to zero.

So, the Lagrange Multipliers tell us the improvement in fit we could expect for each of these constrained parameters. Large statistics essentially tell us there is some source of covariance we are neglecting. We can get a list of all the possible changes to the model using `modindices()` where we enter the fitted model as an argument. We can use the optional argument `sort = TRUE` to display the potential modifications with the largest impact first.

```
# Full list of modification indices (Lagrange Multiplier)
modindices(sem2.fit, sort = TRUE)
```

```
##           lhs op   rhs     mi    epc sepc.lv sepc.all sepc.nox
## 19   populism =~  px06 45.150  0.646   0.447    0.316    0.316
## 43       px07 ~~  px10 37.558  0.113   0.113    0.152    0.152
## 27      pa30r ~~  px06 27.289  0.093   0.093    0.140    0.140
## 36      pa35r ~~  px06 21.820 -0.093  -0.093   -0.127   -0.127
## 28      pa30r ~~  px07 19.353 -0.066  -0.066   -0.103   -0.103
## 32      pa32r ~~  px06 15.618  0.081   0.081    0.111    0.111
## 21   populism =~  px10 15.043 -0.162  -0.112   -0.122   -0.122
## 24 xenophobia =~ pa35r 13.965 -0.119  -0.126   -0.114   -0.114
## 23 xenophobia =~ pa32r 12.703  0.120   0.127    0.111    0.111
## 40       px06 ~~  px07 10.999 -0.167  -0.167   -0.197   -0.197
## 20   populism =~  px07 10.214 -0.208  -0.144   -0.124   -0.124
## 41       px06 ~~  px10  8.143 -0.073  -0.073   -0.094   -0.094
## 26      pa30r ~~ pa35r  7.480  0.055   0.055    0.100    0.100
## 39      pa35r ~~  id02  5.284 -0.032  -0.032   -0.060   -0.060
## 34      pa32r ~~  px10  4.187 -0.029  -0.029   -0.045   -0.045
## 29      pa30r ~~  px10  3.810 -0.024  -0.024   -0.042   -0.042
## 25      pa30r ~~ pa32r  3.535 -0.041  -0.041   -0.074   -0.074
## 30      pa30r ~~  id02  3.473  0.023   0.023    0.048    0.048
## 37      pa35r ~~  px07  0.874  0.016   0.016    0.022    0.022
## 31      pa32r ~~ pa35r  0.785 -0.022  -0.022   -0.036   -0.036
## 33      pa32r ~~  px07  0.468  0.012   0.012    0.017    0.017
## 35      pa32r ~~  id02  0.198  0.006   0.006    0.012    0.012
## 42       px06 ~~  id02  0.092  0.007   0.007    0.010    0.010
## 45       px10 ~~  id02  0.060 -0.003  -0.003   -0.005   -0.005
```

```
## 38      pa35r ~~  px10  0.029 -0.002  -0.002   -0.004   -0.004
## 44       px07 ~~  id02  0.020 -0.002  -0.002   -0.004   -0.004
## 22 xenophobia =~ pa30r  0.014  0.003   0.004    0.004    0.004
```

The table tells us in the first three columns what parameter is meant. For example, `populism =~ px06` means the test is suggesting a cross-loading of the indicator `px06` on populism. The column `mi` tells us the approximate improvement to chi square if we were to allow that parameter to be freely estimated. The larger the value, the greater the improvement. The column after that, `epc` is the expected parameter change. That is an approximation of the parameter estimate without actually having fit the model. So, the factor loading that is now set to zero would likely turn out to be an unstandardized factor loading of about 0.65. However, in practice, the expected change rarely equals the actual change once the model is respecified, so this is just a rough idea. The last columns are variations of standardized expected parameter changes. `sepc.lv` is the "epc" for standardized latent variables, `sepc.all` is the "epc" where both the variables are standardized.[3]

Again, the modification indices are completely exploratory and theory-free. They just tell us where residual correlations are, even if they might make zero sense from a theoretical point of view. In the case of `populism =~ px06`, where `px06` is the item "There is a dangerous amount of foreigners living in Germany," we might be able to make the case that this expression is closely tied to populist arguments in Germany at the moment.

So, this modification seems justified by theory or at least everyday knowledge. However, even if the modification can be justified, it arguably weakens the rest of the model by calling the validity of the measurements into question. As we discussed in the CFA section, construct validity means the latent variable is strongly linked to its indicators (convergent validity), while its indicators should not be strongly linked to the other latent variables (divergent validity). If we cannot establish this baseline, then how can we be sure we are capturing what we intend to in the latent variables? If we cannot establish their validity, then the substantive results of the structural model must also be called into question.

A perhaps more focused way to test the constraints in the model is to look at individual parameters rather than generating a list of all the possible parameters. We can use `lavTestScore()` to investigate this without having to re-estimate the model, which can be convenient. We supply the fitted model as well as the parameters to "add," like this, for example:

```
lavTestScore(sem2.fit, add = "xenophobia =~ pa32r")
```

```
## $test
##
## total score test:
##
##    test     X2 df p.value
## 1 score 12.703  1       0
##
## $uni
##
## univariate score tests:
##
##                 lhs op rhs     X2 df p.value
## 1 xenophobia=~pa32r ==   0 12.703  1       0
```

From this, we see that adding this cross-loading would improve the chi square by about 12.703 units, which is just what we saw in the table of all modification indices above.

[3] I am not sure what `sepc.nox` is because it is not mentioned in the documentation, https://lavaan.ugent.be/tutorial/modindices.html, but it seems to be essentially identical to `sepc.all`.

Dropping redundant parameters

We know that some fit indices like CFI take the parsimoniousness — in other words, the simplicity — of the model into account by creating a ratio of chi square to degrees of freedom. For two models with the same chi square, the CFI and other comparative measures will favour the one with more degrees of freedom, the simpler of the two ("doing more with less"). So, another (debatable) strategy is to look for freely estimated parameters that do not explain very much covariance in the model. Essentially, we are looking for parameters to set to zero at the loss of some fit, but in the service of gaining degrees of freedom.

These are sometimes called Wald statistics, and they tell us the amount with which chi square would worsen if we set the parameter to zero. If the statistic is small, it means the model fit will only decrease slightly, with the benefit of a degree of freedom extra.

To my knowledge, there is no way to get a table or list of all possible parameters that could be set to zero. Instead, we need to test individual hypotheses using `lavTestWald()` where we give the fitted model along with the parameter we want to constrain. For example, we could check the amount with which the model fit would worsen if we fixed the direct effect of class on xenophobia to zero[4]

```
lavTestWald(sem2.fit, constraints = "gamma1 == 0")
```

```
## $stat
## [1] 38.01741
##
## $df
## [1] 1
##
## $p.value
## [1] 7.011625e-10
##
## $se
## [1] "standard"
```

The output tells us the chi square would worsen by approximately 38.017, and that this worsening of fit is highly significant. In other words, fixing the parameter to zero would significantly reduce fit. Ideally, if we are looking for redundant parameters, we would look for those that did not cause a significant decrease in fit.

[4]Notice this is essentially identical to the chi square difference test where we tested the difference in fit between a more general and a more constrained model.

References

Pischke, Jörn-Steffen. 2007. "Lecture Notes on Measurement Error." http://econ.lse.ac.uk/staff/spischke/ec524/Merr_new.pdf.

Wooldridge, Jeffery. 2009. *Introductory Econometrics: A Modern Approach, 4^{th} Edition.* Mason, Ohio: South-Western Cengage Learning.