Covariance Based Structural Equation Modelling

Joerg Evermann

Faculty of Business Administration
Memorial University of Newfoundland
jevermann@mun.ca

- Basics
  - Covariance and Covariance Algebra
  - Covariance Exercise
  - SEM Models
  - SEM Models in R
  - Identification and Estimation
  - Examples in R
  - Second Order Constructs
  - Formative Indicators
  - Miscellaneous but Important Things
  - Publishing SEM
- Advanced Topics
  - Sample Size and Power
  - Multivariate Normality
  - Categorical Variables
  - Multi-Group Analysis
  - Missing Values
  - Interaction/Moderation
  - Latent Growth Models
  - Bayesian SEM
- Latent Class Analysis (LCA)

Traditionally (Analysis of Variance, Regression)

- Model observed and predicted individual responses
- Minimize the differences by adjusting model parameters (e.g. group means or slope & intercept)

Covariance-based Structural Equations

- Model the observed and predicted covariances between variables
- Minimize the difference by adjusting model parameters

- Covariances are a function of a set of model parameters:

$$\Sigma = \Sigma(\theta)$$

- Knowing the parameter values will reproduce the observed covariances
- Model parameters arise from linear regression equations:

$$y = \gamma x + \zeta$$

- Just as in regression, you can transform non-linear variables first

Covariance Algebra

Covariance

The covariance between two random variables X_1 and X_2 is defined as

$$Cov(X_1, X_2) = E\left[(X_1 - E(X_1))(X_2 - E(X_2))\right]$$

The variance is a special covariance of a variable with itself:

$$Cov(X, X) = E\left[(X - E(X))(X - E(X))\right] = Var(x)$$

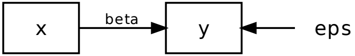
Zero-Centering Values

- Subtract the mean ($= E(X)$) from observed variables
- For zero-centered variables, the covariance simplifies to:

$$Cov(X_1, X_2) = E(X_1 X_2)$$

Example 1

Example 1



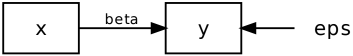
$$y = \beta x + \epsilon$$

$$\begin{aligned} Cov(x,y) &= E\left[xy\right] \\ &= E\left[x(\beta x + \epsilon)\right] \\ &= E\left[\beta xx + x\epsilon\right] \\ &= \beta E\left[xx\right] + E\left[x\epsilon\right] \\ &= \beta Cov(x,x) + Cov(x,\epsilon) \end{aligned}$$

Assuming that $Cov(x,\epsilon) = 0$, i.e. errors do not covary with variables:

$$\begin{aligned} Cov(x,y) = \beta Cov(x,x) &= \beta Var(x) \\ \beta &= \frac{Cov(x,y)}{Var(x)} \end{aligned}$$

Well known result from regression analysis



$$
\begin{aligned}
Cov(y,y) &= E\left[(\beta x+\epsilon)(\beta x+\epsilon)\right] \\
&= \beta\beta Cov(x,x)+\beta Cov(x,\epsilon)+\beta Cov(\epsilon,x)+Cov(\epsilon,\epsilon)
\end{aligned}
$$

Again, we assume that $Cov(x,\epsilon)=0$, i.e. errors do not covary with variables

$$Var(y)=\beta^2 Var(x)+Var(\epsilon)$$

Separation of variance well known from regression

Table: Known Variables

$Cov(x, y)$	$Var(x)$	$Var(y)$

Table: Unknown Variables (Model Parameters)

β	$Var(\epsilon)$

- Model degrees of freedom = number of observed variances/covariances $-$ number of model parameters to estimate

$$df = p^* - t$$

- Typically: $p^* = p(p+1)/2 = p^2/2 + p/2$ where p is the number of observed variables

Estimating Parameters

- Variance and covariance equations allow determination of unknown variables from known variables
- Model implies covariances
- Adjust model parameters β, $\mathit{Var}(\epsilon)$ to match observed covariances

Identifying Equations

Compare the model-implied to the observed covariances, e.g.

$$S = \begin{bmatrix} 1 & \\ 2 & 3 \end{bmatrix} \qquad \Sigma = \begin{bmatrix} Var(x) & \\ \beta Var(x) & \beta^2 Var(x) + Var(\epsilon) \end{bmatrix}$$

From which:

$$1 = Var(x) \tag{1}$$

$$2 = \beta Var(x) \tag{2}$$

$$3 = \beta^2 Var(x) + Var(\epsilon) \tag{3}$$

From equation 1 we have $Var(x) = 1$.

Then, from equation 2 we have $\beta = 2$.

This requires $Var(\epsilon) = -1$ (from equation 3), which is impossible.

There is no single solution satisfying all equations. $\Rightarrow$ The model is over-identified.

A model is

- *overidentified* when $df > 0$,
- *just identified* when $df = 0$, and
- *underidentified* when $df < 0$

Approximate Solutions

Let e.g. *Var*$(\epsilon) = 0$. Then we have from equation 3 that $\beta = \sqrt{3}$. :

$$\hat{\Sigma} = \begin{bmatrix} 1 & \\ \sqrt{3} & 3 \end{bmatrix}$$

With this choice of *Var*(ϵ) we were unable to exactly reproduce the observed covariances and there will be differences between S and $\hat{\Sigma}$.

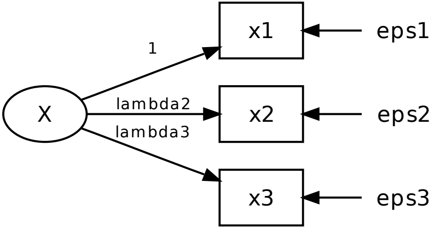
Choose values such that differences between S and $\hat{\Sigma}$ will be minimal.

Example 1 Summary

- $p = 2$ observed variables
- $p^* = 2^2/2 + 2/2 = 3$ observed variances/covariances
- $t = 2$ parameters to estimate
- $df = 3 - 2 = 1$ degrees of freedom ⇒ overidentified ⇒ approximate solution required

Example 2

Example 2



$$
\begin{aligned}
x_1 &= 1X + \epsilon_1 \\
x_2 &= \lambda_2 X + \epsilon_2 \\
x_3 &= \lambda_3 X + \epsilon_3
\end{aligned}
$$

$$
\begin{aligned}
Cov(x_1, x_2) &= E\left[x_1 x_2\right] \\
&= E\left[(X + \epsilon_1)(\lambda_2 X + \epsilon_2)\right] \\
&= E[\lambda_2 XX] + E[X\epsilon_2] \\
&\quad + E[\lambda_1 \epsilon_1 X] + E[\epsilon_1 \epsilon_2] \\
&= \lambda_2 Cov(X, X) = \lambda_2 Var(X)
\end{aligned}
$$

Assumptions:

$$
\begin{aligned}
E[X\epsilon_2] &= Cov(X, \epsilon_2) = 0 \\
E[X\epsilon_1] &= Cov(X, \epsilon_1) = 0 \\
E[\epsilon_1 \epsilon_2] &= Cov(\epsilon_1, \epsilon_2) = 0
\end{aligned}
$$

$$
\begin{aligned}
Cov(x_1, x_3) &= E[x_1 x_3] \\
&= E\left[(X + \epsilon_1)(\lambda_3 X + \epsilon 3)\right] \\
&= \lambda_3 Var(X) \\
Cov(x_2, x_3) &= E[x_2 x_3] \\
&= E\left[(\lambda_2 X + \epsilon_2)(\lambda_3 X + \epsilon_3)\right] \\
&= \lambda_2 \lambda_3 Var(X)
\end{aligned}
$$

$$
\begin{aligned}
Cov(x_1, x_1) = Var(x_1) &= E[x_1 x_1] \\
&= E\left[(X + \epsilon_1)(X + \epsilon_1)\right] \\
&= Var(X) + Var(\epsilon_1) \\
Cov(x_2, x_2) = Var(x_2) &= E[x_2 x_2] \\
&= E\left[(\lambda_2 X + \epsilon_2)(\lambda_2 X + \epsilon_2)\right] \\
&= \lambda_2^2 Var(X) + Var(\epsilon_2) \\
Cov(x_3, x_3) = Var(x_3) &= E[x_3 x_3] \\
&= E\left[(\lambda_3 X + \epsilon_3)(\lambda_3 X + \epsilon_3)\right] \\
&= \lambda_3^2 Var(X) + Var(\epsilon_3)
\end{aligned}
$$

Table: Known Variables

$Cov(x_1, x_2)$	$Cov(x_1, x_2)$	$Cov(x_2, x_3)$
$Var(x_1)$	$Var(x_2)$	$Var(x_3)$

Table: Unknown Variables (Model Parameters)

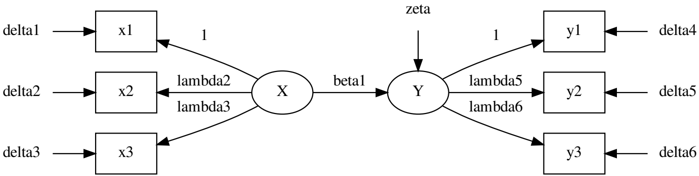
$Var(X)$	λ_2	λ_3
$Var(\epsilon_1)$	$Var(\epsilon_2)$	$Var(\epsilon_3)$

- Variance and covariance equations allow determination of unknown variables from known variables
- Model implies covariances
- Adjust model parameters to match observed covariances

Example 2 Summary

- $p = 3$ observed variables
- $p^* = 3^2/2 + 3/2 = 6$ observed variances/covariances
- $t = 6$ parameters to estimate
- $df = 6 - 6 = 0$ degrees of freedom ⇒ just identified ⇒ exact solution (perfect fit) possible

Covariance Exercise



- How many observed variables are in this model?
- How many variances and covariances will be observed?
- How many parameters do we need to estimate?
- What are the degrees of freedom for this model?
- Is the model overidentified?
- Write this model as a set of regression equations.
- What is the model-implied covariance between x_2 and y_3?

Basic SEModelling

The LISREL Model

Definition (Latent variable)

A variable that is not directly observed or observable. Represents a concept/construct.

Definition (Exogenous latent variable)

A latent variable not explained by the model (”no arrows in”), ξ

Definition (Endogenous latent variable)

A variable explained by the model (”arrows in”), η

Definition (Observed variable)

Also called measured variable or indicator or item. May represent mesurements of questions on survey or experimental measurements.

Figure: Latent variable

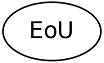
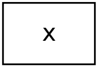
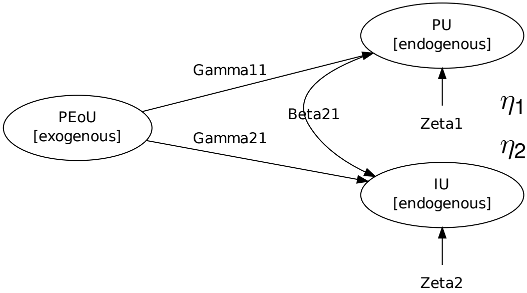


Figure: Observed variable



"Structural Model"



$$\eta_1 = \gamma_{11}\xi_1 + \zeta_1$$
$$\eta_2 = \gamma_{21}\xi_1 + \beta_{21}\eta_1 + \zeta_2$$

$\xi_1 =$ PEoU: Perceived Ease of Use

$\eta_1 =$ PU: Perceived Usefulness

$\eta_2 =$ IU: Intention to Use

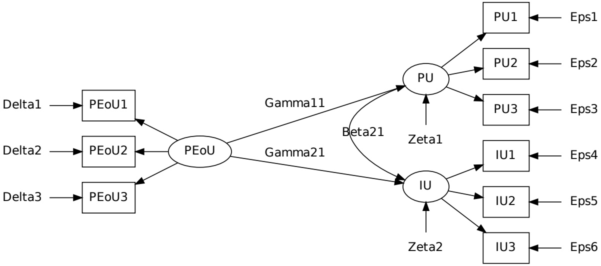
$$
\begin{aligned}
\eta_1 &= \gamma_{11}\xi_1 + \zeta_1 \\
\eta_2 &= \gamma_{21}\xi_1 + \beta_{21}\eta_1 + \zeta_2
\end{aligned}
$$

$$
\begin{bmatrix} \eta_1 \\ \eta_2 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ \beta_{21} & 0 \end{bmatrix} \begin{bmatrix} \eta_1 \\ \eta_2 \end{bmatrix} + \begin{bmatrix} \gamma_{11} \\ \gamma_{21} \end{bmatrix} \begin{bmatrix} \xi_1 \end{bmatrix} + \begin{bmatrix} \zeta_1 \\ \zeta_2 \end{bmatrix}
$$

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

"Measurement Model"

$$x_1 = \lambda_{x1}\xi_1 + \delta_1$$
$$x_2 = \lambda_{x2}\xi_1 + \delta_2$$
$$x_3 = \lambda_{x3}\xi_1 + \delta_3$$



$$y_1 = \lambda_{y1}\eta_1 + \epsilon_1$$
$$y_2 = \lambda_{y2}\eta_1 + \epsilon_2$$
$$y_3 = \lambda_{y3}\eta_1 + \epsilon_3$$
$$y_4 = \lambda_{y4}\eta_2 + \epsilon_4$$
$$y_5 = \lambda_{y5}\eta_2 + \epsilon_5$$
$$y_6 = \lambda_{y6}\eta_2 + \epsilon_6$$

$$\boldsymbol{x} = \boldsymbol{\Lambda}_x\boldsymbol{\xi} + \boldsymbol{\delta}$$
$$\boldsymbol{y} = \boldsymbol{\Lambda}_y\boldsymbol{\eta} + \boldsymbol{\epsilon}$$

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} \lambda_{x1} \\ \lambda_{x2} \\ \lambda_{x3} \end{bmatrix} \begin{bmatrix} \xi_1 \end{bmatrix} + \begin{bmatrix} \delta_1 \\ \delta_2 \\ \delta_3 \end{bmatrix}$$

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \end{bmatrix} = \begin{bmatrix} \lambda_{y1} & 0 \\ \lambda_{y2} & 0 \\ \lambda_{y3} & 0 \\ 0 & \lambda_{y4} \\ 0 & \lambda_{y5} \\ 0 & \lambda_{y6} \end{bmatrix} \begin{bmatrix} \eta_1 \\ \eta_2 \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ \epsilon_4 \\ \epsilon_5 \\ \epsilon_6 \end{bmatrix}$$

$$\boldsymbol{x} = \boldsymbol{\Lambda}_x \boldsymbol{\xi} + \boldsymbol{\delta}$$

$$\boldsymbol{y} = \boldsymbol{\Lambda}_y \boldsymbol{\eta} + \boldsymbol{\epsilon}$$

The "0" coefficients in $\boldsymbol{\Lambda}_y$ are constraints imposed on the model.

The Basic LISREL model

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

Modelling

- Anything that can fit into the basic form of the SEM model is allowed:
  - Multiple latents causing an indicator
  - Single indicator for latent variable
  - Loops (diagonal of $\boldsymbol{B}$)
- This cannot be modelled in the basic LISREL model:
  - Indicators causing other indicators
  - Indicators causing latent variables
  - Exogenous latent (ξ) causing endogenous indicators (y) (or vice versa)
  - Correlated errors between x and y
- The type of relationship expressed by the regression equations is the same between latents and observed as well as between latent and latent variables: Causality

Applying Covariance Algebra to the LISREL Model

$$
\begin{aligned}
Cov(\boldsymbol{x},\boldsymbol{x}) &= E(\boldsymbol{x}\boldsymbol{x}^T) \\
&= E[(\boldsymbol{\Lambda}_x\xi + \delta)(\boldsymbol{\Lambda}_x\xi + \delta)^T] \\
&= E[(\boldsymbol{\Lambda}_x\xi + \delta)(\xi^T\boldsymbol{\Lambda}_x^T + \delta^T)] \\
&= E[\boldsymbol{\Lambda}_x\xi\xi^T\boldsymbol{\Lambda}_x^T + \delta\xi^T\boldsymbol{\Lambda}_x^T + \boldsymbol{\Lambda}_x\xi\delta^T + \delta\delta^T] \\
&= \boldsymbol{\Lambda}_x[E(\xi\xi^T)]\boldsymbol{\Lambda}_x^T + [E(\delta\xi^T)]\boldsymbol{\Lambda}_x^T + \boldsymbol{\Lambda}_x[E(\xi\delta^T)] + E(\delta\delta^T) \\
&= \boldsymbol{\Lambda}_x[\boldsymbol{\Phi}]\boldsymbol{\Lambda}_x^T + Cov(\delta,\xi)\boldsymbol{\Lambda}_x^T + \boldsymbol{\Lambda}_x Cov(\xi,\delta) + \boldsymbol{\Theta}_\delta
\end{aligned}
$$

We assume that the error terms of the indicators are independent of the exogenous concepts:

$$
Cov(\boldsymbol{x},\boldsymbol{x}) = \boldsymbol{\Lambda}_x\boldsymbol{\Phi}\boldsymbol{\Lambda}_x^T + \boldsymbol{\Theta}_\delta
$$

$$
\begin{aligned}
Cov(\boldsymbol{y},\boldsymbol{y}) &= E(\boldsymbol{y}\boldsymbol{y}^T) \\
&= E[(\mathbf{\Lambda}_y\boldsymbol{\eta}+\boldsymbol{\epsilon})(\mathbf{\Lambda}_y\boldsymbol{\eta}+\boldsymbol{\epsilon})^T] \\
&= E[(\mathbf{\Lambda}_y\boldsymbol{\eta}+\boldsymbol{\epsilon})(\boldsymbol{\eta}^T\mathbf{\Lambda}_y^T+\boldsymbol{\epsilon}^T)] \\
&= E((\mathbf{\Lambda}_y\boldsymbol{\eta}\boldsymbol{\eta}^T\mathbf{\Lambda}_y^T+\boldsymbol{\epsilon}\boldsymbol{\eta}^T\mathbf{\Lambda}_y^T+\mathbf{\Lambda}_y\boldsymbol{\eta}\boldsymbol{\epsilon}^T+\boldsymbol{\epsilon}\boldsymbol{\epsilon}^T] \\
&= \mathbf{\Lambda}_y[E(\eta\boldsymbol{\eta}^T)]\mathbf{\Lambda}_y^T+[E(\epsilon\boldsymbol{\eta}^T)]\mathbf{\Lambda}_y^T+\mathbf{\Lambda}_y[E(\eta\boldsymbol{\epsilon}^T)]+E(\epsilon\boldsymbol{\epsilon}^T) \\
&= \mathbf{\Lambda}_y Cov(\eta,\eta)\mathbf{\Lambda}_y^T+Cov(\epsilon,\boldsymbol{\eta}^T)\mathbf{\Lambda}_y^T+\mathbf{\Lambda}_y Cov(\boldsymbol{\eta},\boldsymbol{\epsilon}^T)+\mathbf{\Theta}_\epsilon
\end{aligned}
$$

With similar assumptions as for the $\boldsymbol{x}$, i.e. the errors ϵ are independent of the concepts, we arrive at:

$$Cov(\boldsymbol{y},\boldsymbol{y}) = \mathbf{\Lambda}_y[Cov(\eta,\eta)]\mathbf{\Lambda}_y^T+\mathbf{\Theta}_\epsilon$$

We solve the structural model for $\boldsymbol{\eta}$:

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

With this result for $\boldsymbol{\eta}$ we can calculate the covariance of the $\boldsymbol{\eta}$:

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

With this result for the $Cov(\boldsymbol{\eta},\boldsymbol{\eta})$ we have for the $Cov(\boldsymbol{y},\boldsymbol{y})$:

$$Cov(\boldsymbol{y},\boldsymbol{y}) = \boldsymbol{\Lambda}_y[(\boldsymbol{I}-\boldsymbol{B})^{-1}(\boldsymbol{\Gamma}\boldsymbol{\Phi}\boldsymbol{\Gamma}^T+\boldsymbol{\Psi})(\boldsymbol{I}-\boldsymbol{B})^{-1^T}]\boldsymbol{\Lambda}_y^T+\boldsymbol{\Theta}_\epsilon$$

$$
\begin{aligned}
Cov(\boldsymbol{x}, \boldsymbol{y}) &= E(\boldsymbol{x}\boldsymbol{y}^T) \\
&= E[(\boldsymbol{\Lambda}_x\boldsymbol{\xi} + \boldsymbol{\delta})(\boldsymbol{\Lambda}_y\eta + \boldsymbol{\epsilon})^T] \\
&= E[(\boldsymbol{\Lambda}_x\boldsymbol{\xi} + \boldsymbol{\delta})(\boldsymbol{\eta}^T\boldsymbol{\Lambda}_y^T + \boldsymbol{\epsilon}^T)] \\
&= E[(\boldsymbol{\Lambda}_x\boldsymbol{\xi}\boldsymbol{\eta}^T\boldsymbol{\Lambda}_y^T + \boldsymbol{\delta}\boldsymbol{\eta}^T\boldsymbol{\Lambda}_y^T + \boldsymbol{\Lambda}_x\boldsymbol{\xi}\boldsymbol{\epsilon}^T + \boldsymbol{\delta}\boldsymbol{\epsilon}^T)] \\
&= \boldsymbol{\Lambda}_x E[\boldsymbol{\xi}\boldsymbol{\eta}^T]\boldsymbol{\Lambda}_y^T + E[\boldsymbol{\delta}\boldsymbol{\eta}^T]\boldsymbol{\Lambda}_y^T + \boldsymbol{\Lambda}_x E[\boldsymbol{\xi}\boldsymbol{\epsilon}^T] + E[\boldsymbol{\delta}\boldsymbol{\epsilon}^T] \\
&= \boldsymbol{\Lambda}_x E[\boldsymbol{\xi}\boldsymbol{\eta}^T]\boldsymbol{\Lambda}_y^T
\end{aligned}
$$

We take the derivation of $\boldsymbol{\eta}$ from our structural model and plug it into the formula for the covariances of $\boldsymbol{x}$ and $\boldsymbol{y}$:

$$
\begin{aligned}
Cov(\boldsymbol{x}, \boldsymbol{y}) &= \boldsymbol{\Lambda}_x E[\boldsymbol{\xi}((\boldsymbol{I} - \boldsymbol{B})^{-1}(\boldsymbol{\Gamma}\boldsymbol{\xi} + \boldsymbol{\zeta}))^T]\boldsymbol{\Lambda}_y^T \\
&= \boldsymbol{\Lambda}_x E[\boldsymbol{\xi}(\boldsymbol{\xi}^T\boldsymbol{\Gamma}^T(\boldsymbol{I} - \boldsymbol{B})^{-1^T} + \boldsymbol{\zeta}^T(\boldsymbol{I} - \boldsymbol{B})^{-1^T})]\boldsymbol{\Lambda}_y^T \\
&= \boldsymbol{\Lambda}_x[E(\boldsymbol{\xi}\boldsymbol{\xi}^T)\boldsymbol{\Gamma}^T(\boldsymbol{I} - \boldsymbol{B})^{-1^T} + E(\boldsymbol{\xi}\boldsymbol{\zeta}^T)(\boldsymbol{I} - \boldsymbol{B})^{-1^T}]\boldsymbol{\Lambda}_y^T \\
&= \boldsymbol{\Lambda}_x\boldsymbol{\Phi}\boldsymbol{\Gamma}^T(\boldsymbol{I} - \boldsymbol{B})^{-1^T}\boldsymbol{\Lambda}_y^T
\end{aligned}
$$

Model Parameters

Table: Regression Coefficients

$\boldsymbol{\Gamma}$	Between exogenous and endogenous latents
$\boldsymbol{B}$	Between endogenous latents
$\boldsymbol{\Lambda}_x$	Between exogenous latents and observed variables
$\boldsymbol{\Lambda}_y$	Between endogenous latents and observed variables

Table: Covariances

$\boldsymbol{\Phi}$	Between exogenous latents
$\boldsymbol{\Psi}$	Between errors of endogenous latents
$\boldsymbol{\Theta}_\delta$	Between errors of observed variables (exogenous)
$\boldsymbol{\Theta}_\epsilon$	Between errors of observed variables (endogenous)

Model Parameters

Either

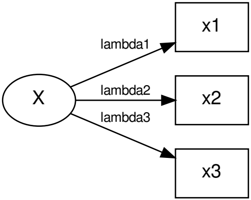
- Free for estimation
- Constrain if known or theoretically motivated
  - to 0 (typical)
  - to any other value
  - to be equal to some other parameter

This applies not only to coefficients but also to exogenous covariances $\mathbf{\Phi}$ and error variances/covariances $\mathbf{\Psi}, \mathbf{\Theta}_\delta, \mathbf{\Theta}_\epsilon$.

Modelling Rules

- Exogenous latents must have a variance
- Exogenous latents typically have free covariances
  - Because their causes are outside our model, they may or may not covary
  - Constrain to 0 (orthogonal) if theoretically justified
- All dependent variables (arrows in) have an error term
  - That part of the variable which our model does not explain
  - Measurement error for observed variables
  - Other causes for endogenous latent variables
  - Constrain to 0 if theoretically justified

Scaling



- Observed variables measured on a $1 \ldots 7$ scale
- Then:
  - Hypothetical X values $1 \ldots 7 \Rightarrow \lambda_i \approx 1$
  - Hypothetical X values $10 \ldots 70 \Rightarrow \lambda_i \approx 0.1$
  - Hypothetical X values $100 \ldots 700 \Rightarrow \lambda_i \approx 0.01$
  - $\ldots$
- Range of hypothetical X values $\sim$ Var(X)
- Scaling either by
  - Fixing one regression coefficient, or
  - Fixing the latent variance

Different SEM Models

- Jöreskog & Sörbom's LISREL
  - Most restrictive
  - Originally implemented in LISREL software
  - LISREL software itself is no longer restricted to this
- Bentler & Weeks' LINEQS
  - Less restrictive
  - Originally implemented in EQS software
  - Allows covariances between *any errors*
- McArdle & McDonald's RAM
  - Least restrictive
  - Implemented in Amos, R (sem), Mx
  - Allows covariances between *any variables*
- Muthén & Muthén
  - Implemented in MPlus, R (lavaan)

LINEQS – Linear Equations (Bentler & Weeks)

- Does not distinguish between latent and observed variables
- Does make a distinction between endogenous and exogenous variables

$$\boldsymbol{\eta} = \boldsymbol{\beta}\boldsymbol{\eta} + \boldsymbol{\gamma}\boldsymbol{\xi}$$

- $\boldsymbol{\eta}$ is the vector of endogenous (latent and observed) variables
- $\boldsymbol{\xi}$ is the vector of exogenous (latent and observed) variables, *including the error variables*,
- $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ are matrices of path coefficients, either constants or parameters.

$$
\begin{aligned}
\boldsymbol{\eta} &= \boldsymbol{\beta}\boldsymbol{\eta} + \gamma\boldsymbol{\xi} \\
\boldsymbol{\eta} - \boldsymbol{\beta}\boldsymbol{\eta} &= \boldsymbol{\gamma}\boldsymbol{\xi} \\
(\boldsymbol{I} - \beta)\boldsymbol{\eta} &= \boldsymbol{\gamma}\boldsymbol{\xi} \\
\boldsymbol{\eta} &= (\boldsymbol{I} - \beta)^{-1}\gamma\boldsymbol{\xi} \\
\boldsymbol{C} = E(\eta\eta)^{T} &= E\left[(\boldsymbol{I} - \beta)^{-1}\gamma\boldsymbol{\xi}\right]\left[(\boldsymbol{I} - \beta)^{-1}\gamma\boldsymbol{\xi}\right]^{T} \\
&= (\boldsymbol{I} - \beta)^{-1}E\left[\gamma\boldsymbol{\xi}\boldsymbol{\xi}^{T}\right]\gamma^{T}(\boldsymbol{I} - \beta)^{-1^{T}} \\
&= (\boldsymbol{I} - \beta)^{-1}\gamma\boldsymbol{\Phi}\gamma^{T}(\boldsymbol{I} - \beta)^{-1^{T}} \\
\boldsymbol{C}_{obs} = E(\eta\eta_{obs}^{T}) &= \boldsymbol{J}(\boldsymbol{I} - \boldsymbol{B})^{-1}\boldsymbol{\Gamma}\Phi\Gamma^{T}(\boldsymbol{I} - \boldsymbol{B})^{-1^{T}}\boldsymbol{J}^{T}
\end{aligned}
$$

with

$$
\boldsymbol{B} = \begin{pmatrix} \beta & 0 \\ 0 & 0 \end{pmatrix} \qquad \text{and} \qquad \boldsymbol{\Gamma} = \begin{pmatrix} \gamma \\ \boldsymbol{I} \end{pmatrix}
$$

RAM — McArdle & McDonald

- Does not distinguish between observed and latent variables
- Does not distinguish between exogenous and endogenous variable

$$\boldsymbol{\nu} = \boldsymbol{A}\boldsymbol{\nu} + \boldsymbol{\mu}$$

- $\boldsymbol{\nu}$ is a vector of random variables
- $\boldsymbol{\mu}$ is a vector of random error variables.
- $\boldsymbol{A}$ is a matrix of (directed) path coefficients

$$
\begin{aligned}
\boldsymbol{\nu} &= \boldsymbol{A}\boldsymbol{\nu} + \boldsymbol{\mu} \\
\boldsymbol{\nu} - \boldsymbol{A}\boldsymbol{\nu} &= \boldsymbol{\mu} \\
(\boldsymbol{I} - \boldsymbol{A})\boldsymbol{\nu} &= \boldsymbol{\mu} \\
\boldsymbol{\nu} &= (\boldsymbol{I} - \boldsymbol{A})^{-1}\boldsymbol{\mu} \\
\boldsymbol{C} = E(\boldsymbol{\nu}\boldsymbol{\nu}^{T}) &= \left[(\boldsymbol{I} - \boldsymbol{A})^{-1}\boldsymbol{\mu}\right]\left[(\boldsymbol{I} - \boldsymbol{A})^{-1}\boldsymbol{\mu}\right]^{T} \\
&= (\boldsymbol{I} - \boldsymbol{A})^{-1} E\left[\boldsymbol{\mu}\boldsymbol{\mu}^{T}\right](\boldsymbol{I} - \boldsymbol{A})^{-1^{T}} \\
&= (\boldsymbol{I} - \boldsymbol{A})^{-1}\boldsymbol{S}(\boldsymbol{I} - \boldsymbol{A})^{-1^{T}} \\
\boldsymbol{C}_{obs} = E(\boldsymbol{\nu}\boldsymbol{\nu}_{obs}^{T}) &= \boldsymbol{J}(\boldsymbol{I} - \boldsymbol{A})^{-1}\boldsymbol{S}(\boldsymbol{I} - \boldsymbol{A})^{-1^{T}}\boldsymbol{J}^{T}
\end{aligned}
$$

Muthén & Muthén

$$\begin{aligned} \boldsymbol{y}_i &= \nu + \boldsymbol{\Lambda}\eta_i + \boldsymbol{K}\boldsymbol{x}_i + \epsilon_i \\ \eta_i &= \alpha + \boldsymbol{B}\eta_i + \boldsymbol{\Gamma}\boldsymbol{x}_i + \zeta_i \end{aligned}$$

- η is a vector of latent variables
- $\boldsymbol{x}$ is a vector of independent variables
- ν is a vector of measurement intercepts
- $\boldsymbol{\Lambda}$ is a matrix of factor loadings
- $\boldsymbol{K}$ is a matrix of regression coefficients
- α is vector of latent intercepts
- $\boldsymbol{B}$ is a matrix of regression coefficients
- $\boldsymbol{\Gamma}$ is a matrix regression coefficients

Let’s do some modelling :-)

The R Statistical System

- General platform for statistical computations
- Programmed using the R programming language (similar to S-Plus)
- Additional packages for various purposes, including SEM
- Packages are installed once and must be loaded whenever needed
- Open-source, free software
- Multi-language & multi-platform

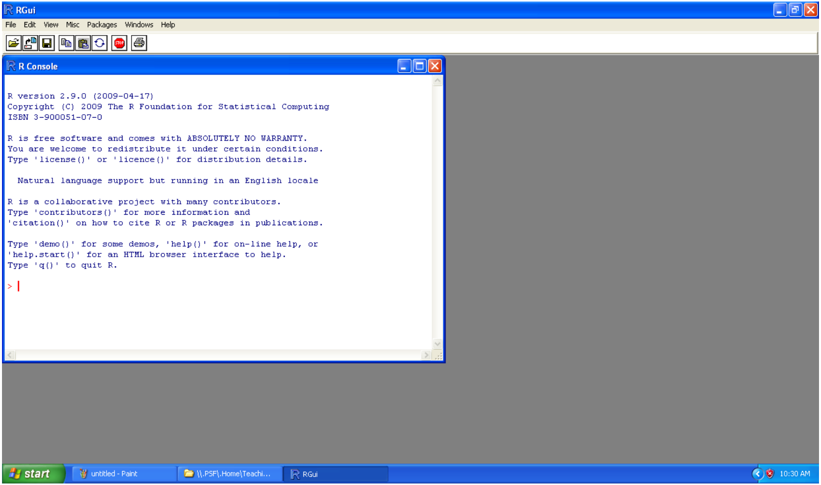
Other SEM Software

- Commercial
  - LISREL
  - EQS
  - Amos
  - MPlus
  - SAS (PROC CALIS)
- Free (closed source)
  - Mx
- Free (open source)
  - sem package for R
  - lavaan package for R
  - OpenMx package for R

Note: The list is not meant to be exhaustive

Installing R

- Download the R version for your operating system (Windows, MacOS, Linux, . . .) and your language from `www.r-project.org`
- Install R as per your operating system
- Launch R as per your operating system



```
R version 2.9.0 (2009-04-17)
Copyright (C) 2009 The R Foundation for Statistical Computing
ISBN 3-900051-07-0

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> 
```

Installing Packages

We require the following packages

- lavaan
- mvnormtest
- norm
- matrixcalc
- Amelia
- polycor
- dprep

To *install* these into your R installation type the following in the R console:

```
> install.packages(c("lavaan", "mvnormtest",
"matrixcalc", "Amelia", "dprep", "norm",
"polycor"))
```

And press the “return” or “enter” key.
R will ask you for a site to download these packages from. *It does not matter which site you pick*.

Setting up the R session

At the beginning of your R session you should

- Set the working directory for R, where you put your data and model files
- Load the required packages

For this, type the following into the R console:

```
> setwd("c:\\path\\to\\directory")
> library(lavaan)
> library(mvnormtest)
> library(matrixcalc)
> library(Amelia)
> library(polycor)
> library(dprep)
> library(norm)
```

And press the “return” or “enter” key after each line.

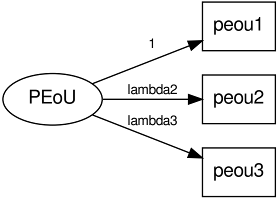
SEM Modelling in R

- Use a text editor (WordPad, TextEdit, . . .) to write model specification files in clear text
- Three basic types of specification

Regressions	`y ~ x1 + x2 + x3`
Covariances	`a ~~ b`
Reflective latent with indicators	`X =~ x1 + x2 + x3`

- Error variances are automatically added to all dependent variables
- Free variances and covariances are automatically added to exogenous latent variables
- Latent variables are automatically scaled by fixing the first indicator path to 1
- Covariances specified on dependent variables (reflective observed, endogenous latent) represent error covariances or disturbances
- Covariances specified on exogenous latent variables represent latent variables covariances.

A Simple Example



Model provided as `model1.txt`
Open and edit with any text editor

Reading the Model

Reading the model file:

```
model1 <- readLines("model1.txt")
```

Printing the model

```
model1
```

R-Tip: `model1` is now a variable in your R workspace

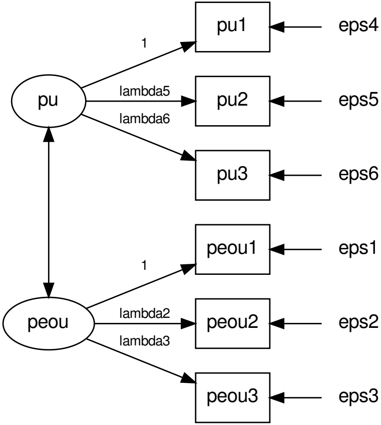
```
peou =~ peou1 + peou2 + peou3
```

All variables for which we don’t have data are assumed to be latent

Automatically added the

- Error variances ($\mathit{diag}(\Theta)$)
- Exogenous latent variances (Φ)
- Scaling constraint

A CFA Example



Model provided as `model2.txt`

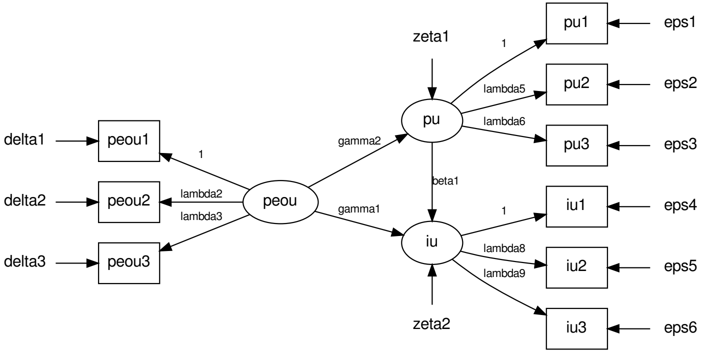
Open and edit with any text editor

```
peou =~ peou1 + peou2 + peou3
pu =~ pu1 + pu2 + pu3
```

Automatically added the

- Latent variances & covariances (Φ)
- Error variances (*diag*(Θ))
- Scaling constraints

A Structural Example



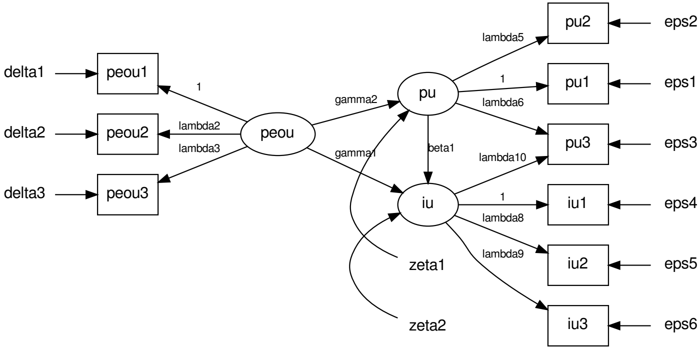
Model provided as `model3.txt`
Open and edit with any text editor

```
peou =~ peou1 + peou2 + peou3
pu =~ pu1 + pu2 + pu3
iu =~ iu1 + iu2 + iu3
pu ~ peou
iu ~ pu + peou
```

Automatically added the

- Latent (co-)variances (Φ)
- Error variances ($diag(\Theta_\delta), diag(\Theta_\epsilon)$)
- Endogenous latent disturbances ($diag(\Psi)$)
- Scaling constraints

Multiple Latents Causing an Indicator



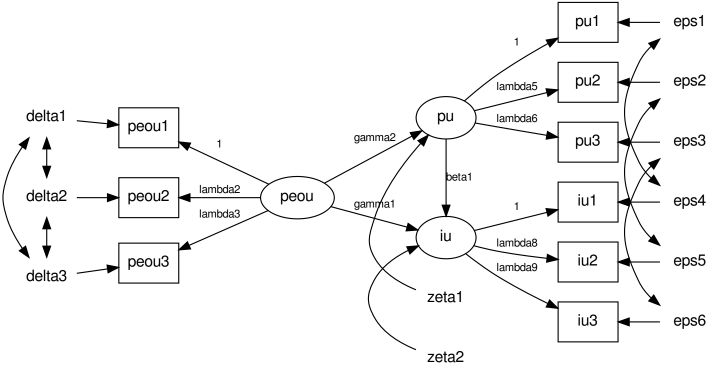
Model provided as `model4.txt`
Open and edit with any text editor

```
peou =~ peou1 + peou2 + peou3
pu =~ pu1 + pu2 + pu3
iu =~ iu1 + iu2 + iu3 + pu3
pu ~ peou
iu ~ pu + peou
```

Automatically added the

- Latent (co-)variances (Φ)
- Error variances ($diag(\Theta_\delta), diag(\Theta_\epsilon)$)
- Endogenous latent disturbances ($diag(\Psi)$)
- Scaling constraints

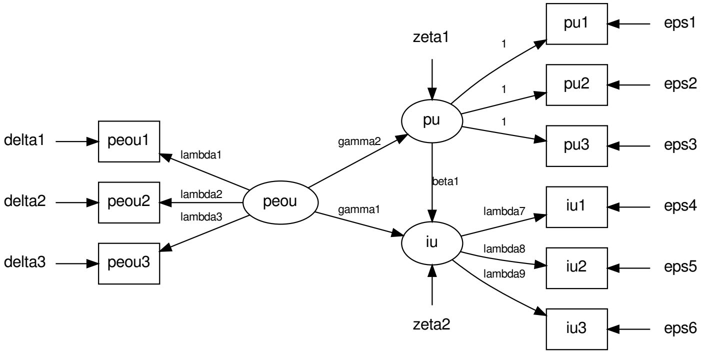
Adding (Error) Covariances



Model provided as `model5.txt`
Open and edit with any text editor

```
peou =~ peou1 + peou2 + peou3
pu =~ pu1 + pu2 + pu3
iu =~ iu1 + iu2 + iu3 + pu3
pu ~ peou
iu ~ peou + pu
pu1 ~~ iu1
pu2 ~~ iu2
pu3 ~~ iu3
peou1 ~~ peou2
peou1 ~~ peou3
peou2 ~~ peou3
```

Fixing (and Freeing) Parameters

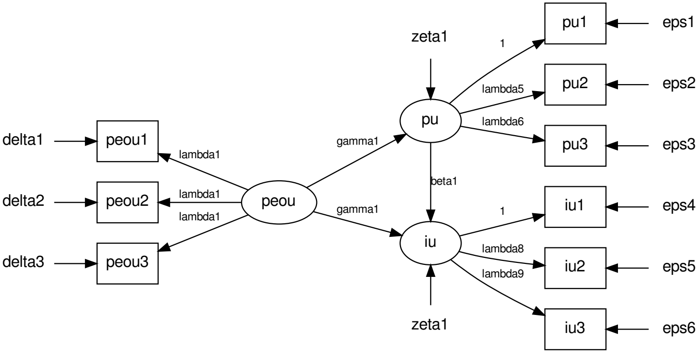


Scale PEOU by fixing variance. Also fix error variances for $\delta_1, \delta_2, \delta_3$.

Model provided as `model6.txt`
Open and edit with any text editor

```
peou =~ NA*peou1 + peou2 + peou3   | Override default
pu =~ 1*pu1 + 1*pu2 + 1*pu3        | Constrain regression
iu =~ NA*iu1 + iu2 + iu3           | Override default
pu ~ peou                          |
iu ~ peou + pu                     |
peou ~~ 2*peou                     | Constrain variance
peou1 ~~ 0.5*peou1                 | Constrain error var.
peou2 ~~ 0.4*peou2                 | Constrain error var.
peou3 ~~ 0.6*peou3                 | Constrain error var.
```

Equality Constraints and Labels

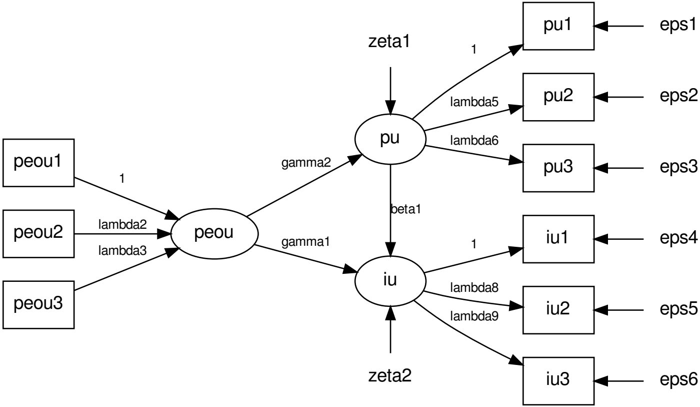


Model provided as `model7.txt`
Open and edit with any text editor

```
peou =~ lambda1*peou1 + lambda1*peou2 + lambda1*peo
pu =~ pu1 + pu2 + pu3
iu =~ iu1 + iu2 + iu3
peou ~~ peou
pu ~ beta1*peou
iu ~ beta1*peou + pu
pu ~~ equal("iu~~iu")*pu
```

- Constrain PEOU loadings (λ) to be identical
- Constrain regressions (γ) from PU to IU to be identical to that from PEOU to IU
- Constrain the endogenous "errors" to have equal variance (ψ)

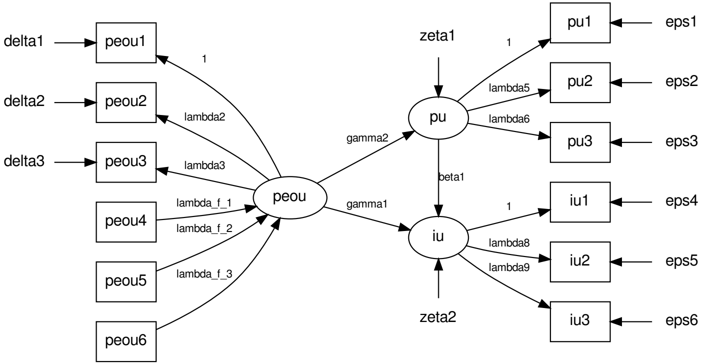
Formative "Indicators"



Model provided as `model8.txt`
Open and edit with any text editor

```
peou ~ 1*peou1 + peou2 + peou3
pu =~ pu1 + pu2 + pu3
iu =~ iu1 + iu2 + iu3
pu ~ peou
iu ~ peou + pu
```

MIMIC Models



Model provided as `model9.txt`
Open and edit with any text editor

```
peou =~ peou1 + peou2 + peou3
peou ~ peou4 + peou5 + peou6
pu =~ pu1 + pu2 + pu3
iu =~ iu1 + iu2 + iu3
pu ~ peou
iu ~ peou + pu
```

Model Identification
& Parameter Aliasing

Identification

Model-implied covariances depend on model parameters:

$$\boldsymbol{\Sigma} = \boldsymbol{\Sigma}(\boldsymbol{\theta})$$

Assume that

$$\boldsymbol{S} = \boldsymbol{\Sigma}(\boldsymbol{\theta})$$

Can we find (one or more) values for $\boldsymbol{\theta}$?

Model Parameters

Table: Coefficients

$\boldsymbol{\Gamma}$	Between exogenous and endogenous latents
$\boldsymbol{B}$	Between endogenous latents
$\boldsymbol{\Lambda}_x$	Between exogenous latents and observed variables
$\boldsymbol{\Lambda}_y$	Between endogenous latents and observed variables

Table: Covariances

$\boldsymbol{\Phi}$	Between exogenous latents
$\boldsymbol{\Psi}$	Between errors of endogenous latents
$\boldsymbol{\Theta}_\delta$	Between errors of observed variables (exogenous)
$\boldsymbol{\Theta}_\epsilon$	Between errors of observed variables (endogenous)

Identification Rules (1)

- Model is identified, if all parameters are identified
- Parameter is identified, if it is possible to find (one or more) solutions from known data (***S***)
    - Model is overidentified if there is no single, exact solution
    - Model constraints prevent such an exact solution
- If values of two parameters depend on each other, they are aliased
- A model with more free parameters than data points (number of unique entries in the covariance matrix) is generally unidentified
    - (Although there may be specific values of θ for which the model is identified)

Identification Rules (Structural Model)

Null-B rule	Endogenous latent variables do not affect each other
Recursive rule	No loops in the relationships among endogenous variables Uncorrelated latent errors (disturbances)

These rules are sufficient, not necessary

Identification Rules (2)

3-indicator rule	three indicators per latent non-correlated errors each indicator caused by exactly one latent
2-indicator rule (A)	two or more exogenous latents two indicators per latent non-correlated errors each indicator caused by exactly one latent non-zero exogenous latent correlations
2-indicator rule (B)	two or more exogenous latents two indicators per latent non-correlated errors each indicator caused by exactly one latent exogenous latents directly or indirectly correlated

These rules are sufficient, not necessary

Identification Rules (3)

2-step rule	Check identification of latent-observed ("measurement") model Check identification of latent-latent ("structural") model sufficient
MIMIC Rule	Latent causes at least two observed variables Latent is "caused by" at least one observed variable necessary

Model Estimation

- Covariances are a function of a set of model parameters:

$$\Sigma = \Sigma(\theta)$$

- Knowing the parameter values will reproduce the observed covariances

1. Take an observed covariance matrix $\boldsymbol{S}$
2. Compare to model implied covariance matrix $\boldsymbol{\Sigma}$
3. Adjust model parameters (and thus $\boldsymbol{\Sigma}$) for closer fit
4. Repeat as necessary

Estimation requires only the covariance matrix, not the raw data

Comparing $\boldsymbol{S}$ and $\boldsymbol{\Sigma}$

- Generalized Least Squares (GLS)
- Weighted Least Squares (WLS)/Asymptotic Distribution Free (ADF)
- *Maximum Likelihood (ML)*
- Robust Maximum Likelihood (MLM)
- Full-information ML (FIML/Raw ML)

All assume multivariate normal data

Likelihood

"Normal" probability is the probability of an observation given a distribution:

$$P(x|\Theta)$$

Likelihood is the probability of a distribution characterized by parameters Θ, given an observation x:

$$L(\Theta|x)$$

The *likelihood function* is the set of proportional probability density functions:

$$L(\Theta|x) = \alpha \times p(x|\Theta)$$

To get rid of the proportionality constant α, one considers likelihood ratios:

$$LR = \frac{L(\Theta|x_1)}{L(\Theta|x_2)} = \frac{P(x_1|\Theta)}{P(x_2|\Theta)}$$

ML Estimator

Probability (and Likelihood) of observing $\boldsymbol{S}$ as a sample of population with $\boldsymbol{\Sigma}$ is given by Wishart distribution:

$$p(S|\Theta) = W(S, \Sigma(\Theta), n) = \frac{e^{-\frac{n}{2}trace(S\Sigma(\Theta)^{-1})}}{|\Sigma(\Theta)|^{n/2}}C$$

Probability (and Likelihood) of observing perfect fit, i.e. $S = \Sigma$:

$$p(S|\Theta) = W(S, S, n) = \frac{e^{-\frac{n}{2}trace(SS^{-1})}}{|S|^{n/2}}C$$

Therefore, the Likelihood Ratio (LR) is:

$$
\begin{aligned}
LR &= \frac{W(S, \Sigma(\Theta), n)}{W(S, S, n)} \\
&= \frac{e^{-\frac{n}{2}trace(S\Sigma(\Theta)^{-1})}}{|\Sigma(\Theta)|^{n/2}} C * \frac{|S|^{n/2}}{e^{-\frac{n}{2}trace(SS^{-1})}} \frac{1}{C} \\
&= e^{-\frac{n}{2}trace(S\Sigma(\Theta)^{-1})} |\Sigma(\Theta)|^{-\frac{n}{2}} e^{\frac{n}{2}trace(SS^{-1})} |S|^{-\frac{n}{2}}
\end{aligned}
$$

Taking the log gives the Log-Likelihood-Ration (LLR):

$$
LLR = -\frac{n}{2}trace(S\Sigma^{-1}) - \frac{n}{2}log|\Sigma| + \frac{n}{2}trace(SS^{-1}) + \frac{n}{2}log|S|
$$

Because $\mathit{trace}(SS^{-1}) = \mathit{trace}(I) = p$ where p is the number of observed variables, we can write

$$LLR = -\frac{n}{2}\mathit{trace}(S\Sigma^{-1}) - \frac{n}{2}\mathit{log}|\Sigma| + \frac{n}{2}p + \frac{n}{2}\mathit{log}|S|$$

Multiplying by $-\frac{2}{n}$ gives the ML *fit function*:

$$F = -\frac{2}{n}LLR = \mathit{trace}(S\Sigma^{-1}) + \mathit{log}|\Sigma| - \mathit{log}|S| - p$$

Other Fit Functions / Estimators

Generalized Least Squares (GLS):

$$\begin{aligned} F_{GLS} &= \frac{1}{2}\frac{n-1}{n} trace\left[(S^{-1}(S-\Sigma))(S^{-1}(S-\Sigma))^T\right] \\ &= \frac{1}{2}\frac{n-1}{n} trace\left[(I-S^{-1}\Sigma)(I-S^{-1}\Sigma)^T\right] \end{aligned}$$

Unweighted Least Squares (ULS):

$$F_{ULS} = \frac{1}{2}\frac{n-1}{n} trace\left[(S-\Sigma)(S-\Sigma)^T\right]$$

Weighted Least Squares (WLS) / Asymptotically Distribution Free (ADF):

$$F_{WLS} = F_{ADF} = (\vec{S}-\vec{\Sigma})^T W^{-1}(\vec{S}-\vec{\Sigma})$$

Adjusting the Model and $\boldsymbol{\Sigma}$

Table: Coefficients

$\boldsymbol{\Gamma}$	Between exogenous and endogenous latents
$\boldsymbol{B}$	Between endogenous latents
$\boldsymbol{\Lambda}_x$	Between exogenous latents and observed variables
$\boldsymbol{\Lambda}_y$	Between endogenous latents and observed variables

Table: Covariances

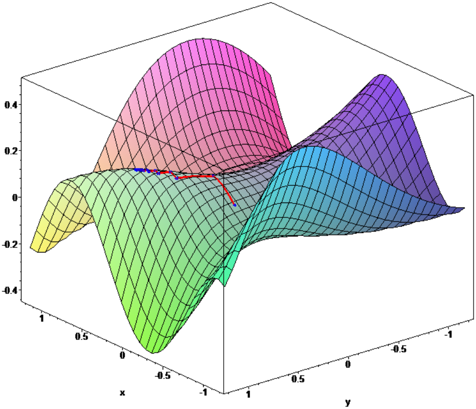
$\boldsymbol{\Phi}$	Between exogenous latents
$\boldsymbol{\Psi}$	Between errors of endogenous latents
$\boldsymbol{\Theta}_\delta$	Between errors of observed variables (exogenous)
$\boldsymbol{\Theta}_\epsilon$	Between errors of observed variables (endogenous)

Gradient Method

- Differentiate F w.r.t. $\mathbf{\Theta}$

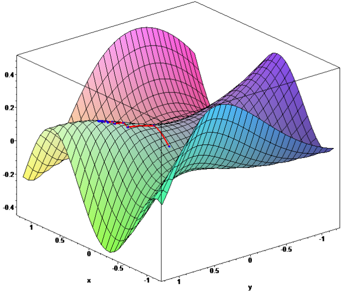
$$\nabla F = \left[\frac{\partial F}{\partial \Theta_i} \right]$$

- Make changes to the parameter $\mathbf{\Theta}$ with the greatest derivative



Modification Indices / Lagrange Multipliers

- What if you could into "forbidden" directions in the gradient method?
- Forbidden directions are those parameters that are fixed in the model (e.g. to 0)
- MI calculated from gradient of fit function:



$$MI = \frac{n-1}{2} \left(\frac{\partial F}{\partial \Theta}\right)^{T} \left[E\left(\frac{\partial^2 F}{\partial \Theta \partial \Theta^{T}}\right)\right]^{-1} \left(\frac{\partial F}{\partial \theta}\right)$$

Model Fit

The ML fit function is a function of the Log-Likelihood-Ratio (LLR):

$$F = -\frac{2}{n} LLR = trace(S\Sigma^{-1}) + log|\Sigma| - log|S| - p$$

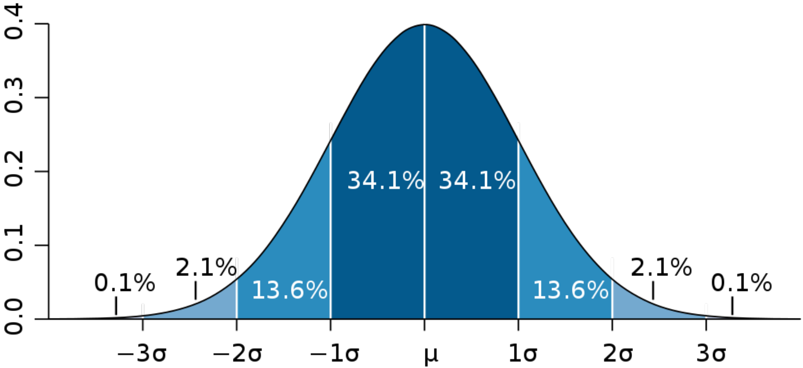
Any LLR multiplied by -2 is χ^2 distributed:

$$\chi^2 \sim nF = -2 \times LLR = n\left[trace(S\Sigma^{-1}) + log|\Sigma| - log|S| - p\right]$$

With $df = p^* - t$ where $p^* = p(p+1)/2$ is the number of observed variances/covariances and t is the number of parameters in Θ that are estimated. Further

$$E(\chi^2_{df}) = df$$

Statistical Significance Tests



The χ^2 Test

- Assumes multivariate normal data
  - Robust (Satorra-Bentler) correction for non-normal data
  - Failure to fit may also be due to non-normal data
- The only real statistical test (i.e. with known α and β levels)
- Tests *overall* model fit, not individual constraints
  - Tests the set of constraints imposed by the model, e.g. paths that are *not* in the model and equality constraints
  - Free paths in the model (and their parameter estimates) are not what is being tested

Model Comparisons — The χ^2 Difference Test

Recall that:

$$\chi^2 = -2 log \left(\frac{L_{model}}{L_{perfectModel}} \right)$$

Hence:

$$\begin{aligned}
\Delta\chi^2 = \chi^2_2 - \chi^2_1 &= -2 \left[log \left(\frac{L_{model2}}{L_{perfectModel}} \right) - log \left(\frac{L_{model1}}{L_{perfectModel}} \right) \right] \\
&= -2 log \left[\frac{\frac{L_{model2}}{L_{perfectModel}}}{\frac{L_{model1}}{L_{perfectModel}}} \right] \\
&= -2 log \left(\frac{L_{model2}}{L_{model1}} \right)
\end{aligned}$$

Model Comparisons — The χ^2 Difference Test

Recall that:

$$df = p - t$$

Hence:

$$\begin{aligned}\Delta df &= df_2 - df_1 \\ &= (p_2 - t_2) - (p_1 - t_1)\end{aligned}$$

Assuming that $p_1 = p_2$:

$$\Delta df = t_1 - t_2$$

- Difference in the number of parameters to estimate
- Difference in the number of constraints on the model

Model Comparisons — The χ^2 Difference Test

- For nested models only
  - Models are nested, if one is a more constrained form of the other
- Rather than comparing a model against a perfectly fitting model, the null hypothesis in this case is that there are no differences between the two models in the way they fit the data.
- A significant $\Delta\chi^2$ would indicate that one model fits the data better than the other.

Model Test

- χ^2 test only shows that a model fits the observed covariances
- It does not show the model is the true model
  - There may be multiple models that fit the data
  - There may be models that are *covariance equivalent*, i.e. with identical $\boldsymbol{\Sigma}$
- Use SEM to test and compare alternative theories
  - Which should be theoretically motivated

Absolute Fit Indices

$$GFI = 1 - \frac{trace([\mathbf{\Sigma}^{-1}(\boldsymbol{S} - \mathbf{\Sigma})]^2)}{trace[(\mathbf{\Sigma}^{-1}S)^2]}$$

$$AGFI = 1 - \frac{n(n+1)}{2df}(1 - GFI)$$

$$RMSEA = \sqrt{max(\frac{F}{df} - \frac{1}{N-1}, 0)}$$

$$BIC = -2log(L) + t \times ln(N)$$

$$AIC = -2Log(L) + 2t \qquad log(L) \text{ = Log-Likelihood}$$

- (Controversial) Cutoff heuristics for these indices
- No statistical test (i.e. unknown α and β levels)
- Useful for comparing models that fit by the χ^2 criteria

$n =$ Number of exogenous latent vars, $N =$ Number of obs (sample size), $t =$ Number of free parameters to estimate

Relative Fit Indices

$$NFI = \frac{T_B - T_T}{T_B} \qquad BL86 = \frac{\frac{T_B}{df_B} - \frac{T_T}{df_t}}{\frac{T_B}{df_B}}$$

$$TLI = \frac{\frac{T_B}{df_B} - \frac{T_T}{df_t}}{\frac{T_B}{df_B} - 1} \qquad BL89 = \frac{T_B - T_T}{T_B - df_T}$$

$$BLI = \frac{(T_B - df_B) - (T_T - df_T)}{T_B - df_B}$$

$$CFI = 1 - \frac{max\left[(T_T - df_T), 0\right]}{max\left[(T_T - df_T), (T_B - df_B), 0\right]}$$

- Compare a baseline model to a target model
    - Independence model with (uncorrelated) observed variables only. Only $diag(\Sigma)$ estimable
- T is the χ^2 test statistic for the models

Fit and Model Specification

- No connection between degree of mis-specification of model and χ^2 misfit
  - A small modelling mistake may lead to a large misfit
  - A large modelling mistake may lead to a small misfit
  - "Almost fitting" does not imply "Almost correct"
- Modification indices are ill-suited
  - MI are equivalent to single degree of freedom χ^2 difference tests (i.e. of a single constraint)
  - Assume model is substantially well-specified and ill-fit is due to a single constraint

Testing Parameters

Each parameter estimate has a standard error associated with it:

$$SE_\theta = \left(\frac{2}{(N-1)}\right)\left(E\left[\frac{\partial^2 F_{ML}}{\partial\theta\partial\theta^T}\right]\right)^{-1}$$

The ratio of estimate and error is normally distributed and can be tested for significance (i.e. difference from 0):

$$z \sim \frac{\theta}{SE_\theta}$$

Explained Variance

- Explained variance in observed variables
- Note that σ denotes elements of Σ

$$r_i^2 = 1 - \frac{var(\delta_i)}{\sigma_{ii}}$$

Standardizing Coefficients

- Parameter estimates depend on unit of measurement of variables
- Difficult to compare parameters between different variables and measurement scales
- Standardize using variance estimates

$$\lambda_{ij}^{s} = \lambda_{ij} \left(\frac{\sigma_{jj}}{\sigma_{ii}} \right)^{1/2}$$

$$\beta_{ij}^{s} = \beta_{ij} \left(\frac{\sigma_{jj}}{\sigma_{ii}} \right)^{1/2}$$

$$\gamma_{ij}^{s} = \gamma_{ij} \left(\frac{\sigma_{jj}}{\sigma_{ii}} \right)^{1/2}$$

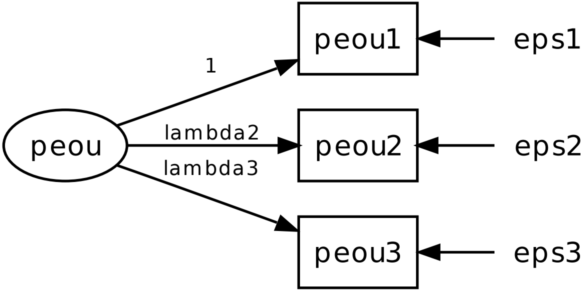
Traditional Validity and Reliability

These apply to a factor model only!

$$
\begin{aligned}
AVE &= \frac{\sum_i \lambda_i^2}{\sum_i \lambda_i^2 + \sum_i var(\epsilon_i)} \\
CR &= \frac{(\sum_i \lambda_i)^2}{(\sum_i \lambda_i)^2 + \sum_i var(\epsilon_i)} \\
&= \frac{\sum_i \sum_{j \neq i} \lambda_i \lambda_j + \sum_i \lambda_i^2}{\sum_i \sum_{j \neq i} \lambda_i \lambda_j + \sum_i \lambda_i^2 + \sum_i var(\epsilon_i)} \\
AVE &\leq CR
\end{aligned}
$$

Example 1

```
          peou1     peou2     peou3
peou1 1.2461590 0.9659238 0.8048996
peou2 0.9659238 0.8108234 0.6567238
peou3 0.8048996 0.6567238 0.5872959
```



Estimation in R

Read the model file:

```
model1 <- readLines(’model1.txt’)
```

Read the data file:

```
data1 <- read.csv(’dataset1.csv’)
```

Call the `sem` function to estimate the model:

```
sem1 <- sem(model1, data=data1)
```

Print the summary:

```
summary(sem1)
```

```
Lavaan (0.4-9) converged normally after 27 iterations

  Number of observations                         10000

  Estimator                                         ML
  Minimum Function Chi-square                    0.000
  Degrees of freedom                                 0
  P-value                                        0.000

                   Estimate  Std.err  Z-value  P(>|z|)
Latent variables:
  peou =~
    peou1             1.000
    peou2             0.816    0.002  338.240    0.000
    peou3             0.680    0.002  277.956    0.000

Variances:
    peou1             0.062    0.001   46.123    0.000
    peou2             0.023    0.001   30.153    0.000
    peou3             0.040    0.001   54.264    0.000
    peou              1.184    0.018   67.146    0.000
```

Comments on Example 1

- Note how R scales the exogenous latent by fixing the first indicator λ to 1
- Unstandardized regression coefficients
- Default ML estimation
- Just identified model (6 unknown model parameters, 6 known covariance data points $\Rightarrow$ 0 df), always perfect fit ($\chi^2 \approx 0, p \approx 1$)

- `model1, data1, sem1` are objects in the R workspace
- Naming of objects is arbitrary
- Most R objects have a `summary()` function
- Use `ls()` to list all objects in your R workspace
- Use `rm()` to remove an object from your R workspace

Alternative: Scaling by fixing LV variance

Call the `sem` function to estimate the model:

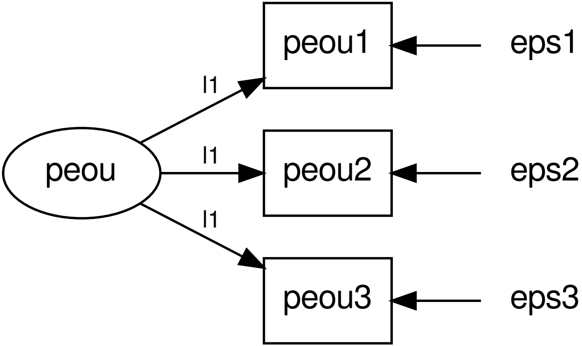
```
sem1 <- sem(model1, data=data1, std.lv=T)
```

Print the summary:

```
summary(sem1)
```

```
Lavaan (0.4-9) converged normally after 40 iterations

  Number of observations                         10000

  Estimator                                         ML
  Minimum Function Chi-square                    0.000
  Degrees of freedom                                 0
  P-value                                        0.000

                   Estimate  Std.err  Z-value  P(>|z|)
Latent variables:
  peou =~
    peou1             1.088    0.008  134.293    0.000
    peou2             0.888    0.006  137.270    0.000
    peou3             0.740    0.006  131.870    0.000

Variances:
    peou1             0.062    0.001   46.123    0.000
    peou2             0.023    0.001   30.153    0.000
    peou3             0.040    0.001   54.264    0.000
    peou              1.000
```

Constraints and Nested Models



```
peou =~ l1*peou1 + l1*peou2 + l1*peou3
```

Estimation in R

Read the model file:

```
model1a <- readLines('model1a.txt')
```

Call the `sem` function to estimate the model:

```
sem1a <- sem(model1a, data=data1, sdt.lv=T)
```

Print the summary:

```
summary(sem1a)
```

```
Lavaan (0.4-9) converged normally after 33 iterations

  Number of observations                          10000

  Estimator                                          ML
  Minimum Function Chi-square                  7962.058
  Degrees of freedom                                  2
  P-value                                         0.000

                   Estimate  Std.err  Z-value  P(>|z|)
Latent variables:
  peou =~
    peou1             0.900    0.006  140.883    0.000
    peou2             0.900    0.006  140.883    0.000
    peou3             0.900    0.006  140.883    0.000

Variances:
    peou1             0.125    0.002   62.690    0.000
    peou2             0.001    0.001    0.674    0.500
    peou3             0.084    0.002   55.873    0.000
    peou              1.000
```

Compute $\Delta\chi^2$:

```
delta.chisq <- 7962.058 - 0
```

Compute Δdf:

```
delta.df <- 2 - 0
```

Calculate the p-value:

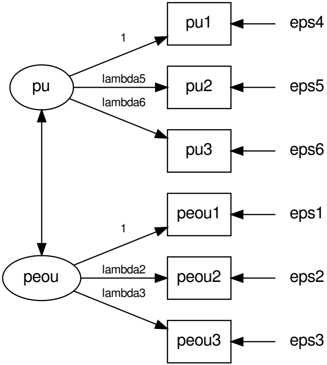
```
p.value <- pchisq(delta.chisq, delta.df,
lower.tail=F)
```

And print it:

```
p.value
[1] 0
```

Example 2

```
        peou1   peou2   peou3     pu1      pu2      pu3
peou1 1.26542 0.98873 0.82375 0.27293 0.197255 0.244091
peou2 0.98873 0.83407 0.67587 0.22307 0.161148 0.201114
peou3 0.82375 0.67587 0.60289 0.18595 0.133825 0.166997
pu1   0.27293 0.22307 0.18595 0.32311 0.055070 0.070570
pu2   0.19726 0.16115 0.13382 0.05507 0.129530 0.050269
pu3   0.24409 0.20111 0.16700 0.07057 0.050269 0.102491
```



Estimation in R

Read the model file:

```
model2 <- readLines(’model2.txt’)
```

Read the covariance matrix:

```
data2.cov <- read.table(’dataset2.cov.txt’,
row.names=1)
```

Call the `sem` function to estimate the model:

```
sem2 <- sem(model2,
sample.cov=as.matrix(data2.cov),
sample.nobs=10000)
```

Print the summary, including standardized estimates:

```
summary(sem2, standardized=T)
```

```
Lavaan (0.4-9) converged normally after 64 iterations

  Number of observations                         10000

  Estimator                                         ML
  Minimum Function Chi-square                    5.934
  Degrees of freedom                                 8
  P-value                                        0.655

                   Estimate  Std.err  Z-value  P(>|z|)   Std.lv  Std.all
Latent variables:
  peou =~
    peou1             1.000                               1.098    0.976
    peou2             0.821    0.002  346.650    0.000    0.901    0.986
    peou3             0.684    0.002  283.612    0.000    0.750    0.966
  pu =~
    pu1               1.000                               0.279    0.491
    pu2               0.719    0.018   39.537    0.000    0.201    0.558
    pu3               0.898    0.019   46.478    0.000    0.251    0.783

Covariances:
  peou ~~
    pu                0.272    0.007   40.271    0.000    0.889    0.889

Variances:
    peou1             0.060    0.001   46.757    0.000    0.060    0.048
    peou2             0.023    0.001   31.356    0.000    0.023    0.027
    peou3             0.040    0.001   54.845    0.000    0.040    0.066
    pu1               0.245    0.004   66.613    0.000    0.245    0.759
    pu2               0.089    0.001   64.703    0.000    0.089    0.689
    pu3               0.040    0.001   42.974    0.000    0.040    0.386
    peou              1.205    0.018   67.308    0.000    1.000    1.000
    pu                0.078    0.003   24.193    0.000    1.000    1.000
```

Comments on Example 2

- Calling `sem()` with the covariance matrix and the number of observations is useful when the full data set is not available
- Two ways of standardizing coefficients:
  1. `Std.lv` scales so that latent variable variances are 1
  2. `Std.all` scales so that latent and observed variances are 1 (*completely standardized*)
- R automatically covaries the exogenous latent variables
- 13 Model parameters, and $\frac{6^2}{2} + \frac{6}{2} = 21$ covariance datapoints ⇒ 8 df
- No significant differences between S and Σ (p-value = 0.6546)

Fit Indices

Print the summary, including fit measures:

```
summary(sem2, fit.measures=T)
```

```
Lavaan (0.4-9) converged normally after 64 iterations

  Number of observations                         10000

  Estimator                                         ML
  Minimum Function Chi-square                    5.934
  Degrees of freedom                                 8
  P-value                                        0.655

Chi-square test baseline model:

  Minimum Function Chi-square                62830.028
  Degrees of freedom                                15
  P-value                                        0.000

Full model versus baseline model:

  Comparative Fit Index (CFI)                    1.000
  Tucker-Lewis Index (TLI)                       1.000

Loglikelihood and Information Criteria:

  Loglikelihood user model (H0)             -24203.060
  Loglikelihood unrestricted model (H1)     -24200.093

  Number of free parameters                         13
  Akaike (AIC)                               48432.120
  Bayesian (BIC)                             48525.855
  Sample-size adjusted Bayesian (BIC)        48484.542

Root Mean Square Error of Approximation:

  RMSEA                                          0.000
  90 Percent Confidence Interval          0.000  0.010
  P-value RMSEA <= 0.05                          1.000

Standardized Root Mean Square Residual:
```

Comments

- Baseline model is the independence model with uncorrelated observed variables

```
fpeou1 =~ peou1
fpeou2 =~ peou2
fpeou3 =~ peou3
fpu1 =~ pu1
fpu2 =~ pu2
fpu3 =~ pu3
peou1 ~~ 0*peou1
peou2 ~~ 0*peou2
peou3 ~~ 0*peou3
pu1 ~~ 0*pu1
pu2 ~~ 0*pu2
pu3 ~~ 0*pu3
```

```
fpeou1 ~~ 0*fpeou2
fpeou1 ~~ 0*fpeou3
fpeou1 ~~ 0*fpu1
fpeou1 ~~ 0*fpu2
fpeou1 ~~ 0*fpu3
fpeou2 ~~ 0*fpeou3
fpeou2 ~~ 0*fpu1
fpeou2 ~~ 0*fpu2
fpeou2 ~~ 0*fpu3
fpeou3 ~~ 0*fpu1
fpeou3 ~~ 0*fpu2
fpeou3 ~~ 0*fpu3
fpu1 ~~ 0*fpu2
fpu1 ~~ 0*fpu3
fpu2 ~~ 0*fpu3
```

Additional Fit Indices

Load additional functions:

```
source('sem.GFI.R')
```

Ask for the GFI:

```
sem.GFI(sem2)
```

```
[1] 0.9996806
```

Explained Variance

Ask for the r^2 values:

```
summary(sem2, rsquare=T)
```

```
Lavaan (0.4-9) converged normally after 64 iterations

...

R-Square:

    peou1                 0.952
    peou2                 0.973
    peou3                 0.934
    pu1                   0.241
    pu2                   0.311
    pu3                   0.614
```

SEM Diagnostics

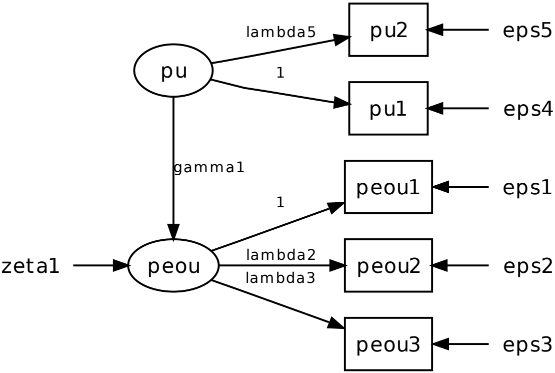
SEM Diagnostics

- Covariance Matrix
  - Are there systematic covariances not in the model?
- Modification Indices
  - How much would the fit (or χ^2) improve if you could change a fixed parameter?
- Residuals
  - How much do the actual and model-implied covariances differ ($\boldsymbol{S} - \boldsymbol{\Sigma}$)

Modification Indices

Modification Indices

```
          peou1     peou2     peou3       pu1       pu2
peou1 1.2758051 0.9895795 0.8216456 0.2991939 0.2158660
peou2 0.9895795 0.8287384 0.6697273 0.2403090 0.1725382
peou3 0.8216456 0.6697273 0.5963789 0.2046119 0.1457410
pu1   0.2991939 0.2403090 0.2046119 1.1353142 0.9570547
pu2   0.2158660 0.1725382 0.1457410 0.9570547 0.9458446
```



Estimation in R

Read the model file:

```
model10 <- readLines(’model10.txt’)
```

Read the data file:

```
data10 <- read.csv(’dataset10.csv’)
```

Call the `sem` function to estimate the model:

```
sem10 <- sem(model10, data=data10)
```

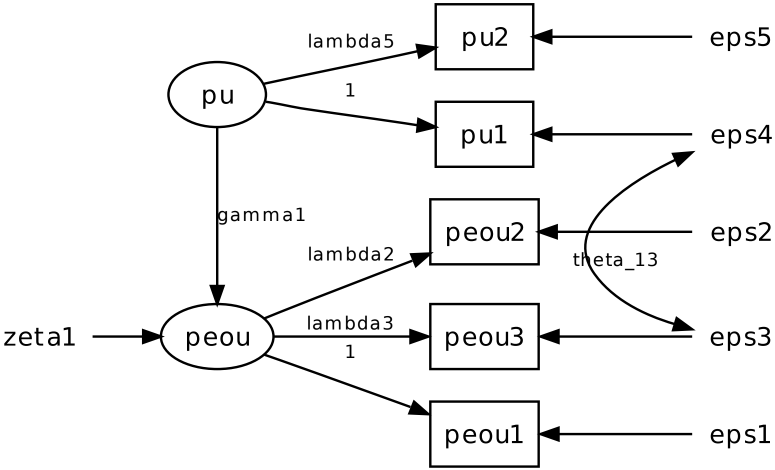
Print the summary, including modification indices:

```
summary(sem10, modindices=T)
```

```
Modification Indices:

     lhs op   rhs    mi    epc sepc.lv sepc.all
1   peou =~ peou1    NA     NA      NA       NA
2   peou =~ peou2 0.000  0.000   0.000    0.000
3   peou =~ peou3 0.000  0.000   0.000    0.000
4   peou =~   pu1    NA     NA      NA       NA
5   peou =~   pu2    NA     NA      NA       NA
6     pu =~ peou1 0.101  0.001   0.001    0.001
7     pu =~ peou2 5.059 -0.004  -0.004   -0.005
8     pu =~ peou3 6.029  0.004   0.005    0.006
9     pu =~   pu1    NA     NA      NA       NA
10    pu =~   pu2 0.000  0.000   0.000    0.000
11 peou1 ~~ peou1 0.000  0.000   0.000    0.000
12 peou1 ~~ peou2 6.029  0.026   0.026    0.025
13 peou1 ~~ peou3 5.059 -0.016  -0.016   -0.018
14 peou1 ~~   pu1 0.547 -0.001  -0.001   -0.001
15 peou1 ~~   pu2 1.258  0.001   0.001    0.001
16 peou2 ~~ peou2 0.000  0.000   0.000    0.000
17 peou2 ~~ peou3 0.101  0.002   0.002    0.003
18 peou2 ~~   pu1 2.577 -0.001  -0.001   -0.001
19 peou2 ~~   pu2 0.748  0.001   0.001    0.001
20 peou3 ~~ peou3 0.000  0.000   0.000    0.000
21 peou3 ~~   pu1 8.097  0.003   0.003    0.003
22 peou3 ~~   pu2 5.526 -0.002  -0.002   -0.003
23   pu1 ~~   pu1 0.000  0.000   0.000    0.000
24   pu1 ~~   pu2    NA     NA      NA       NA
25   pu2 ~~   pu2 0.000  0.000   0.000    0.000
26  peou ~~  peou 0.000  0.000   0.000    0.000
27  peou ~~    pu    NA     NA      NA       NA
28    pu ~~    pu 0.000  0.000   0.000    0.000
29  peou  ~    pu 0.000  0.000   0.000    0.000
30    pu  ~  peou    NA     NA      NA       NA
```

Comments

- This model does not fit the data ($p = 0.0310$)
- Modification Indices (MI) reflect the improvement that can be expected by freeing a path
- Expected Parameter Change (EPC) indicates by how much that freed parameter is likely to change at the new optimum
- MI are *ceteris paribus* changes
- MI assume an otherwise correct model
- Large MI are often those associated with error covariances
  - Cheap way of improving model fit: Reduce a single residual
  - Difficulty in theoretically justifying free parameters
- Do not only attend to single MI, but consider them in context


pu2
eps5
lambda5
pu
1
pu1
eps4
gamma1
peou2
eps2
lambda2
theta_13
zeta1
peou
lambda3
peou3
eps3
1
peou1
eps1

Estimation in R

Read the model file:

```
model10 <- readLines('model10a.txt')
```

Call the `sem` function to estimate the model:

```
sem10 <- sem(model10, data=data10)
```

Print the summary, including modification indices:

```
summary(sem10, modindices=T)
```

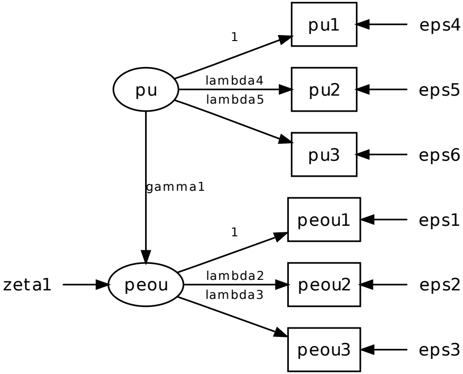
```
Lavaan (0.4-9) converged normally after 78 iterations

  Number of observations                         10000

  Estimator                                         ML
  Minimum Function Chi-square                    2.537
  Degrees of freedom                                 3
  P-value                                        0.469

                   Estimate  Std.err  Z-value  P(>|z|)

...

Covariances:
  peou3 ~~
    pu1               0.003    0.001    2.841    0.004

...
```

Comments

- The freed error covariance changed slightly, as was expected from the EPC
- The error covariance is significantly different from 0
- The MI have changed, based on this new model
- "Chasing after" MI will not typically lead to the true model

Residuals

```
       peou1  peou2   peou3    pu1     pu2     pu3
peou1 1.4030 1.0890 0.88980 1.2660 0.12180 0.14890
peou2 1.0890 0.9103 0.72720 1.0330 0.10650 0.11590
peou3 0.8898 0.7272 0.63090 0.8046 0.05993 0.06351
pu1   1.2660 1.0330 0.80460 2.4660 1.17800 1.48900
pu2   0.1218 0.1065 0.05993 1.1780 1.02900 1.21000
pu3   0.1489 0.1159 0.06351 1.4890 1.21000 1.59200
```



Estimation in R

Read the model file:

```
model11 <- readLines(’model11.txt’)
```

Read the data file:

```
data11 <- read.csv(’dataset11.csv’)
```

Call the `sem` function to estimate the model:

```
sem11 <- sem(model11, data=data11,
mimic=’EQS’)
```

Print the summary:

```
summary(sem11)
```

Note: By default, `lavaan` mimics the MPlus software and multiplies the covariance matrix $\boldsymbol{S}$ by $\frac{N-1}{N}$. This can be turned off, so that the results are closer to those of EQS or LISREL.

```
lavaan (0.4-9) converged normally after 62 iterations

  Number of observations                           100

  Estimator                                         ML
  Minimum Function Chi-square                  224.958
  Degrees of freedom                                 8
  P-value                                        0.000
```

Ask for residuals:

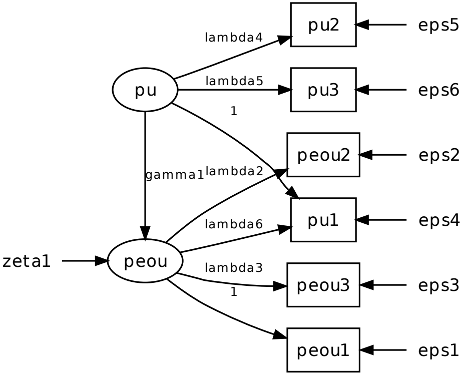
```
residuals(sem11)
```

Or, subtract the fitted values, i.e. the estimated $\hat{\boldsymbol{\Sigma}}$ from the sample covariance:

```
cov(data11) - fitted.values(sem11)$cov
```

```
$cov
      peou1  peou2  peou3  pu1    pu2    pu3
peou1  0.000
peou2  0.000  0.000
peou3  0.000  0.000  0.000
pu1    1.086  0.886  0.684  0.000
pu2   -0.025 -0.013 -0.038 -0.006  0.000
pu3   -0.035 -0.035 -0.059 -0.002  0.001  0.000
```

- Look for large or systematic residuals
- Similar to MI, do not consider residuals individually but consider them in context



Estimation in R

Read the model file:

```
model11 <- readLines(’model11a.txt’)
```

Call the `sem` function to estimate the model:

```
sem11 <- sem(model11, data=data11,
mimic=’EQS’)
```

Print the summary:

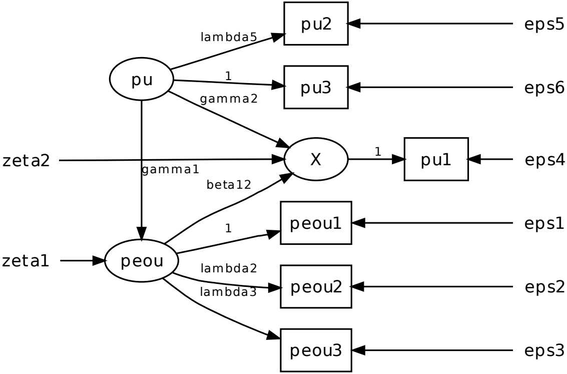
```
summary(sem11)
```

```
peou =~ peou1 + peou2 + peou3 + pu1
pu =~ pu1 + pu2 + pu3
peou ~ pu
```

```
Lavaan (0.4-9) converged normally after 69 iterations

  Number of observations                           100

  Estimator                                         ML
  Minimum Function Chi-square                    5.487
  Degrees of freedom                                 7
  P-value                                        0.601

                   Estimate  Std.err  Z-value  P(>|z|)
Latent variables:
  peou =~
    peou1             1.000
    peou2             0.819    0.022   36.447    0.000
    peou3             0.666    0.023   28.445    0.000
    pu1               0.848    0.035   24.534    0.000
  pu =~
    pu1               1.000
    pu2               0.880    0.036   24.491    0.000
    pu3               1.126    0.038   29.954    0.000

Regressions:
  peou ~
    pu                0.102    0.106    0.959    0.338
```

Is the effect from PEoU to PU1 realistic or theoretically justifiable?

- Another way of addressing the residuals
- Covariance equivalent to Model 11a



Estimation in R

Read the model file:

```
model11 <- readLines('model11b.txt')
```

Call the `sem` function to estimate the model:

```
sem11 <- sem(model11, data=data11,
mimic='EQS')
```

Print the summary:

```
summary(sem11)
```

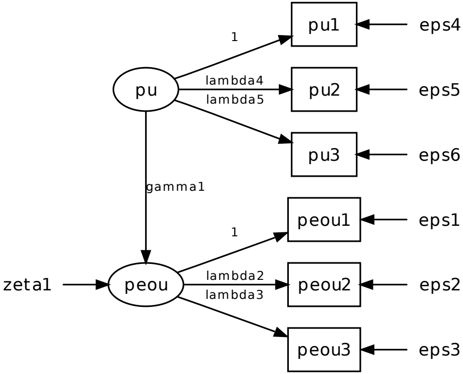
```
peou =~ peou1 + peou2 + peou3
pu =~ pu1 + pu2 + pu3
peou ~ pu
X =~ pu1
X ~ 1*pu + peou
pu1 ~~ 0*pu1
```

```
Lavaan (0.4-9) converged normally after 69 iterations

  Number of observations                            100

  Estimator                                          ML
  Minimum Function Chi-square                     5.487
  Degrees of freedom                                  7
  P-value                                         0.601

                  Estimate  Std.err   Z-value  P(>|z|)
Latent variables:
  X =~
    pu1               1.000

Regressions:
  X ~
    pu                1.000
    peou              0.848     0.035   24.534    0.000

Variances:
    X                 0.077     0.018    4.284    0.000
```

What does this mystery latent variable X represent?

Modeling Methods
Influence

```
       peou1  peou2  peou3    pu1    pu2    pu3
peou1 1.6104 1.3087 1.1147 0.4148 0.4021 0.3576
peou2 1.3087 1.1188 0.9398 0.3728 0.3579 0.3104
peou3 1.1147 0.9398 0.8402 0.3418 0.3293 0.2905
pu1   0.4148 0.3728 0.3418 1.0202 0.9406 1.0126
pu2   0.4021 0.3579 0.3293 0.9406 1.0265 1.0129
pu3   0.3576 0.3104 0.2905 1.0126 1.0129 1.1230
```



Estimation in R

Read the model file:

```
model12 <- readLines('model12.txt')
```

Read the data file:

```
data12 <- read.csv('dataset12.csv')
```

Call the `sem` function to estimate the model:

```
sem12 <- sem(model12, data=data12)
```

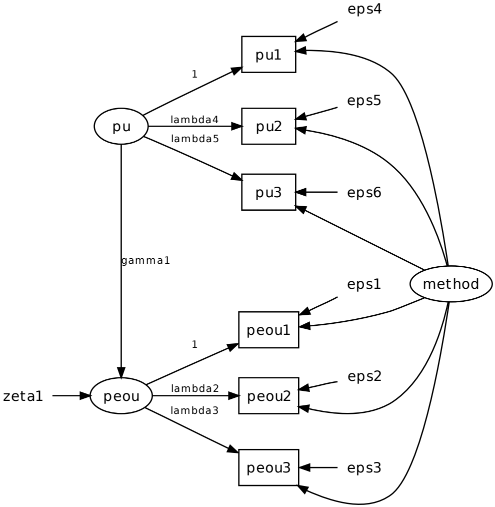
Print the summary:

```
summary(sem12)
```

```
Lavaan (0.4-9) converged normally after 46 iterations

  Number of observations                          100

  Estimator                                        ML
  Minimum Function Chi-square                  16.217
  Degrees of freedom                                8
  P-value                                       0.039
```

Bad fit with regular model

Methods Model



Estimation in R

Read the model file:

```
model12 <- readLines('model12a.txt')
```

Call the `sem` function to estimate the model:

```
sem12 <- sem(model12, data=data12)
```

Print the summary:

```
summary(sem12)
```

```
Lavaan (0.4-9) converged normally after 68 iterations

  Number of observations                             100

  Estimator                                           ML
  Minimum Function Chi-square                      9.047
  Degrees of freedom                                   7
  P-value                                          0.249

...

                   Estimate  Std.err  Z-value  P(>|z|)
Variances:
    method            0.257    0.083    3.112    0.002
```

Better fit with method explicitly modeled

```
peou =~ peou1 + peou2 + peou3
pu =~ pu1 + pu2 + pu3
peou ~ pu
method =~ 1*peou1 + 1*peou2 + 1*peou3 + 1*pu1 + 1*pu2 + 1*pu3
method ~~ 0*pu
```

- Method as exogenous latent
- Method uncorrelated with other exogenous latents
- Method has same effect on all observed variables
- Method has no indicators
- Inclusion of method leads to changes in parameter estimates
- Is it really (and only) a method effect?

Second Order Constructs

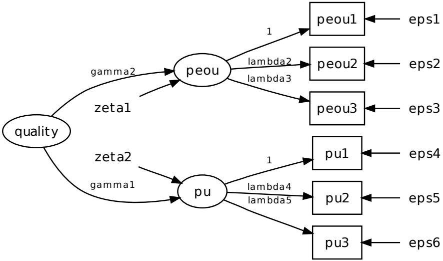
- Theories (conceptual models) may use terminology such as "second order concept", "dimension of", "facet of", "aspect of", etc.
- SEM (statistical model) contains only latent and observed variables

  $\Rightarrow$ Multiple ways of representing conceptual models with statistical models
- Different representations have different implications
- Clear presentation of statistical model is important

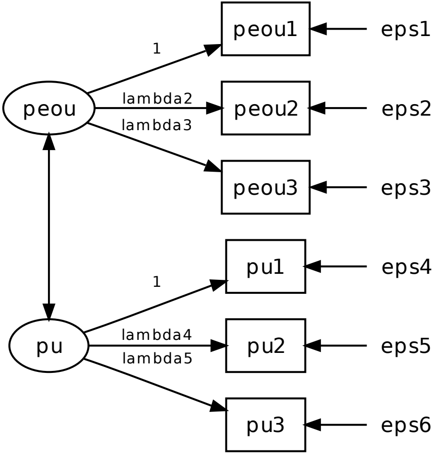
Example

"Quality has two dimensions, Perceived Ease of Use and Perceived Usefulness"

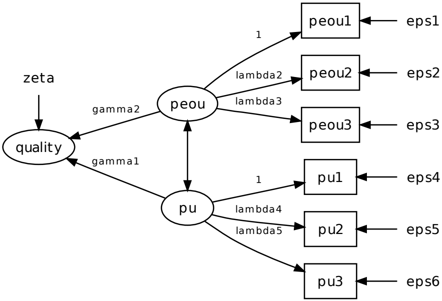
- Quality is distinct from PU and PEoU
- Quality causes PEoU and PU
- Other causes of PEoU and PU (ζ_1, ζ_2)
- Causes of quality are outside of model scope
- Quality is not directly observed
- EoU and U *completely* mediate the effect of quality on indicators



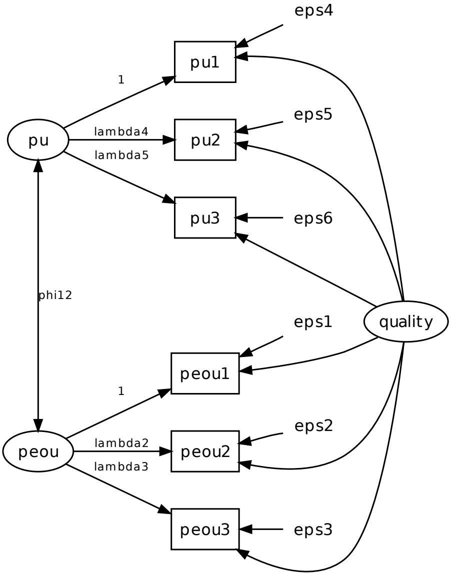
- Quality is nothing but the combination of PEoU and PU
  - There is no separate latent for it
- Causes of PEoU and PU are outside of model scope
- Statistically equivalent to previous model!



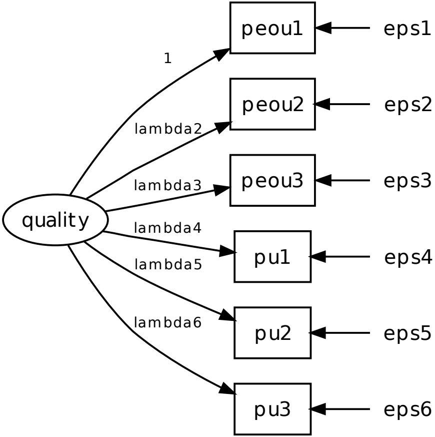
- Quality is distinct from PEoU and PU
- Quality is caused by PEoU and PU
- Quality has other causes (ζ)
  - Or does it? ($var(\zeta) = 0$)
- Causes of PEoU and PU are otuside model scope
- Quality is not directly observed



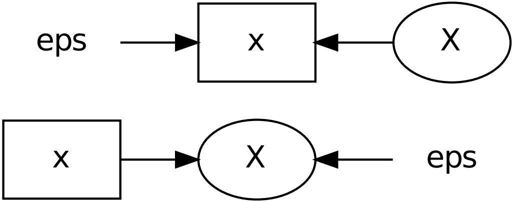
- Quality is distinct from PEoU and PU
- Quality and PEoU and PU are unrelated
- Causes of Quality, PEoU and PU are otuside model scope
- Quality is observable by the same indicators as PEoU and PU
- Same as common methods model ?!



- Quality directly causes the observations
- PEoU and PU do not exist independently of quality
- Causes of Quality are outside model scope
- Why make a conceptual distinction?



Formative Indicators


eps
x
X
x
X
eps

???

Reflective (”normal”) Items

- Latent causes indicators
- Indicators are correlated / covary
  - Latent is common cause
- Dropping one of the (equivalent) indicators does not change the meaning of the latent
- Items have measurement error
- Latent possesses its own meaning
- Items don’t have consequences, but same antecedent

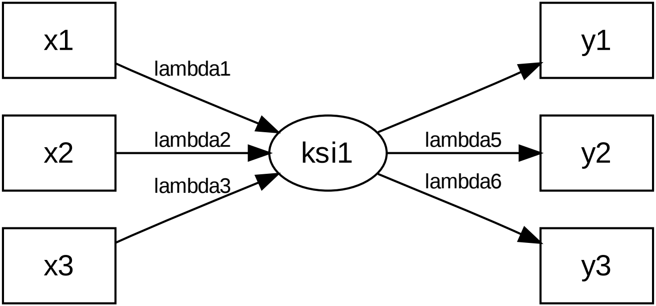
Formative ”Indicators”

- ”Indicators” cause latent
  - Is this measurement or definition?
  - Is this theoretically plausible?
- ”Indicators” need not be correlated
  - They are exogenous
  - There is no common cause
  - Similar to regression, we want maximally independent indicators to avoid multicollinearity

Formative ”Indicators”

- Dropping one of the (*non-equivalent*) ”indicators” changes the meaning of the latent variable
- Error/Residual term associated with latent variable
  - What does the latent error represent?
  - Should it be 0?
- ”Indicators” are assumed to have no measurement error
- Validity and reliability measures (e.g. Cronbach's α, AVE or internal consistency/composite reliability) are not applicable
  - They are based on indicators with common cause

Formative "Indicators"



- MIMIC Model: Multiple Indicators, Multiple Causes

Model13

```
ksi =~ y1 + y2 + y3
ksi ~ x1 + x2 + x3
```

- Model is identified by MIMIC rule (2 or more reflective indicators)
- `x1`, `x2`, `x3` have no error variance

Estimation in R

Read the model file:

```
model13 <- readLines(’model13.txt’)
```

Read the data file:

```
data13 <- read.csv(’dataset13.csv’)
```

Call the `sem` function to estimate the model:

```
sem13 <- sem(model13, data=data13,
mimic=’EQS’)
```

Print the summary:

```
summary(sem13)
```

```
Lavaan (0.4-9) converged normally after 54 iterations

  Number of observations                          1000

  Estimator                                         ML
  Minimum Function Chi-square                    9.683
  Degrees of freedom                                 6
  P-value                                        0.139

                   Estimate  Std.err  Z-value  P(>|z|)
Latent variables:
  ksi =~
    y1                1.000
    y2                0.889    0.006  139.209    0.000
    y3                1.102    0.009  121.619    0.000

Regressions:
  ksi ~
    x1                0.487    0.008   63.073    0.000
    x2                0.648    0.008   85.576    0.000
    x3                0.800    0.008  102.799    0.000

Covariances:
  x1 ~~
    x2                0.004    0.031    0.135    0.893
    x3                0.041    0.031    1.329    0.184
  x2 ~~
    x3               -0.022    0.033   -0.675    0.500

Variances:
    y1                0.039    0.002   16.457    0.000
    y2                0.024    0.002   14.933    0.000
    y3                0.064    0.004   18.079    0.000
    ksi               0.037    0.002   16.166    0.000
    x1                0.918    0.041   22.349    0.000
    x2                1.024    0.046   22.349    0.000
    x3                1.045    0.047   22.349    0.000
```

Compare $\boldsymbol{S}$ with $\hat{\boldsymbol{\Sigma}}$:

```
cov(data13)
```

```
fitted.values(sem13)$cov
```

```
> cov(data13)
            x1           x2          x3        y1        y2        y3
x1 0.917504113  0.004134004  0.04121171 0.4850540 0.4329195 0.5188117
x2 0.004134004  1.023692970 -0.02208616 0.6543297 0.5735043 0.7075315
x3 0.041211713 -0.022086158  1.04541518 0.8356451 0.7480129 0.9390306
y1 0.485053954  0.654329668  0.83564506 1.4039009 1.2137539 1.5038576
y2 0.432919547  0.573504261  0.74801294 1.2137539 1.1038883 1.3381877
y3 0.518811663  0.707531514  0.93903063 1.5038576 1.3381877 1.7207181
> fitted.values(sem13)$cov
   y1     y2     y3     x1     x2     x3
y1  1.404
y2  1.214  1.104
y3  1.504  1.337  1.721
x1  0.482  0.429  0.531  0.918
x2  0.647  0.576  0.713  0.004  1.024
x3  0.842  0.749  0.928  0.041 -0.022  1.045
```

- No measurement error variance on `x1, x2, x3`
- Variances/Covariances among exogenous `x1, x2, x3` equal the observed covariances

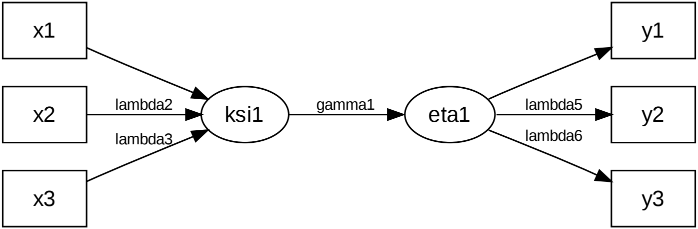
Covariates Treatment (`fixed.x`)

- Formative items are called regression "covariates".
- Maybe treated as

1. Fixed effects
   - Not estimated, fixed to their observed value
   - Default when `mimic='Mplus'`
   - Can be forced with `fixed.x=T`
2. Random effects
   - Variances and covariances are estimated
   - Default when `mimic='EQS'`
   - Can be forced with `fixed.x=F`

MIMIC Models

- MIMIC models are statistically estimable, but difficult to interpret
  - e.g. Responses on a survey cause the latent, which causes other responses?
- Consider MIMIC as ”shorthand” for a model with two latents:



Model13a

`eta =~ y1 + y2 + y3`	Reflective latent
`ksi =~ NA*eta`	Eta is "indicator" of Ksi
`eta ~~ eta`	Free Eta residual variance
`ksi ~~ 0*ksi`	Fix Ksi residual variance
`ksi ~ 1*x1 + x2 + x3`	Regressions on covariates

- Aliasing of λ_i, ζ_1, γ_1, ζ_2 requires constraining a parameter, here λ_1, for identification

Estimation in R

Read the model file:

```
model13 <- readLines(’model13a.txt’)
```

Call the `sem` function to estimate the model:

```
sem13 <- sem(model13, data=data13,
mimic=’EQS’)
```

Print the summary:

```
summary(sem13)
```

```
Lavaan (0.4-9) converged normally after 40 iterations

  Number of observations                            1000

  Estimator                                           ML
  Minimum Function Chi-square                      9.683
  Degrees of freedom                                   6
  P-value                                          0.139

                   Estimate  Std.err  Z-value  P(>|z|)
Latent variables:
  ksi =~
    eta               0.487    0.008   63.073    0.000

Regressions:
  ksi ~
    x1                1.000
    x2                1.330    0.025   53.342    0.000
    x3                1.643    0.029   55.991    0.000

Covariances:
  x1 ~~
    x2                0.004    0.031    0.135    0.893
    x3                0.041    0.031    1.329    0.184
  x2 ~~
    x3               -0.022    0.033   -0.675    0.500
```

Note: Same fit → Covariance-equivalent models

Mis-Specification

- It's a big problem
- Jarvis et al. (2003) found 29% incorrectly specified studies in the top marketing journals (1192 studies, 4 journals, 2 years)
- In a simulation study, the parameter estimates were biased between -100% to 600% compared to true values

Theoretical and Statistical Models

- Theories (conceptual models) deal with many different concepts and relationships, e.g. "concept", "construct", "dimension", etc.
- SEM Models contain only latent and observed variables, and regression relationships
- They are not the same!!
- Statistical models *represent* theories

Translation from Theory to SEM

- Begin with sound theoretical definitions
- Each latent variable should represent a uni-dimensional construct
    - Define the ontology of your ”multi-dimensional constructs”
- Reflective indicators should be equivalent, not just "related to the same idea"
- Each latent variable should immediately cause observations on observed variables (*proximal latents*)
    - Otherwise, use more latent variables
- Regression relationships typically represent causality
    - Assess their plausibility
    - Measurement relationships also represent causality, they are not just ”a set of items (somehow) related to a concept”
- If there are ambiguities, you probably need more latent variables, representing different concepts

Importance of Measurement Instrument

- Factor model offers great parsimony
- Requires great care
- Model fit problems because of
  - Insistence on factor model
  - Careless construction of instrument
  - Neglect of subtle but important relationships between items
- CB-SEM is *not* a combination of factor analysis and regression, it offers much greater flexibility

Publishing SEM

- Detailed and critical examination of theory and existing measurement instruments
- Clear, focused, and specific conceptual definitions
  - A measurable property, not a thing!
  - e.g. not "knowledge", but "depth of knowledge"
  - Construct definition should not consist of relationships to other concepts, leave that for theory development
- Specify a-prior alternative models based on alternative theories
- Publish full statistical model with indicators and error terms (or their estimates)
- Specify how theory/conceptual models are translated to SEM
  - e.g. "Higher order concepts", "dimensions", etc.

- Publish how data are collected
  - Are the estimated error variances plausible?
  - Are the errors independent?
- Pearson or polychoric correlations?
- How is multi-variate normality assessed and, if necessary, addressed?
- Are missign values MCAR or MAR? How are they treated?
  - FIML
  - EM, MI and other imputation methods
  - Bootstrapping
- Publish covariance matrix and sample size
  - Allows validation of research by others
  - Allows extension of research by others
  - Allows alternative explanations to be tested at a later time

- Publish estimation method (ML, GLS, etc.)
- Publish appropriate fit evaluations
  - Include the χ^2 and its df
  - Select carefully from fit indices
- Publish parameter estimates and significance
  - Ensure variances > 0, standardized covariances < 1
- Publish post-hoc model modification based on MI or residuals
  - Each modification requires theoretical justification and plausibility!
  - Explicitly acknowledge the bias in the post-hoc model test

- Careful interpretation of the model, paying attention to
    - Go beyond the model fit
    - Non-significant paths (what does this imply for the tested theory?)
    - Path coefficients differing from
        - Previous studies
        - Different models
        - Different groups in one study
    - These may indicate different meanings of the concepts (*interpretational confounding*)

Covariance-Based SEM and PLS

Objectives

- Covariance-based SEM:
  - Explain observed covariances
- PLS
  - Maximize explained variance, i.e. minimize errors in the endogenous latent variables (ζ)

Common Myths

Myth	**Fact**
SEM does not allow formative models	All SEM packages provide formative modelling
SEM requires normal data	SEM can be bootstrapped just like PLS or use corrections for robust parameter estimates.
SEM requires large sample sizes	Simulation studies have shown SEM estimates to be better than PLS estimates for all sample sizes (Goodhue et al., 2006)

CB-SEM vs. PLS

- SEM provides global model fit criterion
- SEM provides statistical tests

- SEM allows MIMIC models
- SEM allows parameter constraints (fixed values, equality)
- SEM allows explicit measurement error specifications (variance and covariance)

- SEM is conceptually simpler
- SEM is better understood in terms of its statistical properties and problems
- SEM is widely used and has many support options (software, books, journals)

Advanced Topics

Advanced Topics

- Sample size and power
- Multivariate normality
- Categorical variables
- Multi-group analysis
- Missing values
- Interaction/moderation models
- Latent growth models

Sample Size and Power

Sample Size and The χ^2 Test

- Formally, the fit function F (and hence the χ^2 statistic) has a linear dependence on the sample size n.
- For a correct model, the χ^2 statistic is centrally distributed χ^2 ($E = df$), while for an incorrect model, the χ^2 statistic is non-centrally distributed (non-centrality parameter λ, $E = df + \lambda$) [Wilks, 1938]
- Large sample size is *no* reason to disregard χ^2 ill-fit
- Large sample size increases power of model test
  - ”It’s a feature, not a bug”

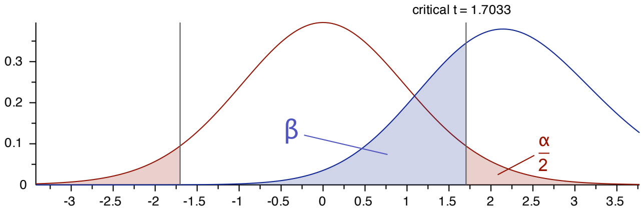
Simulation

	Correct Model	Small Misspecification	Large Misspecification
n=1000	8%	6%	100%
n=10000	4%	59%	100%
n=100000	3%	100%	100%

- Two factor CFA model
- Misspecified by single crossloading at .01 (small), .1 (large)
- Percentage significant χ^2 of 100 simulation runs

Sample Size and Power

- Power calculations require Null- and Alternative Hypotheses (H_0, H_A)
- In simple tests, e.g. equality of means, the observed value is considered as H_A
- In SEM, H_A represents an alternative model
  - Power of the test to distinguish between the two competing models
  - Alternative models rarely given



Note: Power = $1 - \beta$

RMSEA Power

- Power for RMSEA statistic, not the χ^2 test, based on [MacCallum, Browne, Sugawara, 1996]

- Typical recommendation
  - RMSEA $\approx 0.08 \rightarrow$ "Close fit"
  - RMSEA $< 0.05 \rightarrow$ "Good fit"
  - RMSEA $\approx 0.01 \rightarrow$ "Excellent fit"

$$\text{RMSEA} = \sqrt{max\left(\frac{F}{df} - \frac{1}{N-1}, 0\right)}$$

- Test of Close Fit
  - RMSEA under $H_0 = 0.05$
  - RMSEA under $H_A = 0.08$
- Test of Exact Fit
  - RMSEA under $H_0 = 0.00$
  - RMSEA under $H_A = 0.05$
- Test of Not Close Fit
  - RMSEA under $H_0 = 0.05$
  - RMSEA under $H_A = 0.01$

RMSEA Power in R

Load some additional functions:

```
source("rmsea.power.R")
```

Power of a test for ”Close Fit” ($H_A = .08$) with 30 df and sample size 200

```
rmsea.power(0.08, 30, 200)
[1] 0.5848019
```

Power of a test for ”Not Close Fit” ($H_A = .01$) with 5 df and sample size 400

```
rmsea.power(0.01, 5, 400)
[1] 0.2483507
```

Power of a test for ”Exact Fit” ($H_A = .05, H_0 = 0$) with 20 df and sample size 300

```
rmsea.power(.05, 20, 300, 0)
[1] 0.6091227
```

- Power increases with sample size
    - Lower standard errors on estimates
- Power increases with degrees of freedom of the model
    - ”More ways in which the model may not fit”

df	Test	Sample Size		
		100	200	400
5	Close	.127	.199	.335
	Not Close	.081	.124	.248
	Exact	.112	.188	.362
10	Close	.169	.294	.520
	Not Close	.105	.191	.429
	Exact	.141	.266	.541
20	Close	.241	.454	.766
	Not Close	.148	.314	.695
	Exact	.192	.400	.773
50	Close	.424	.769	.981
	Not Close	.261	.608	.969
	Exact	.319	.684	.979
100	Close	.650	.955	1.000
	Not Close	.426	.870	1.000
	Exact	.491	.904	1.000

From MacCallum, Browne, Sugawara (1996)

Sample Size in R

Load some additional functions:

```
source(’sem.minsample.R’)
```

Sample size for a test for ”Close Fit” ($H_A = 0.08$) with df=3, df=30, df=300:

```
sem.minsample(.08, 3)
[1] 2362.5
sem.minsample(.08, 30)
[1] 314.0625
sem.minsample(.08, 300)
[1] 65.625
```

Sample size for a test for ”Not Close Fit” ($H_A = 0.01$) with df=3, df=30, df=300:

```
sem.minsample(.01, 3)
[1] 1762.5
sem.minsample(.01, 30)
[1] 365.625
sem.minsample(.01, 300)
[1] 95.70312
```

- Sample size requirements increase as the df decrease
- Intuitively:
  - Smaller standard errors for estimates needed as there are fewer things about the model that may be wrong

df	Min N for Close Fit	Min N for Not Close Fit
2	3488	2382
4	1807	1426
10	782	750
20	435	474
30	314	366
40	252	307
50	214	268
75	161	210
100	132	178

MacCallum, Browne, Sugawara (1996)

Multivariate Normality

Multivariate Normality

- Non-normality may affect parameter estimates
- Statistical tests (χ^2, $z-$test) assume multivariate normality
- Non-normality negatively biases the standard errors for significance tests, i.e. they will be underestimated
- May lead to significance claims where not significant

- For a CFA model, ML χ^2 is robust if
  1. ξ and ϵ are mutually independent
  2. ϵ are mutually independent of each other
  3. Elements of Φ are freely estimated
  4. Diagonals of Ψ are freely estimated
- For a full SEM model, ML χ^2 is robust if
  1. Independence of exogenous variables, assumed to be uncorrelated
  2. Free estimation of the non-zero elements in Φ

[Savalei, 2008]

MV Normality Tests

- *Skewness and Kurtosis (Mardia’s coefficients) (1970)*
- Cox-Small test (1978)
- *Multivariate Shapiro-Wilks test (1982)*
- Smith-Jain test (1988)
- Baringhaus-Henze test (1988)
- E-test (2005)

Mardia's Coefficients

Load the `dprep` library:

```
library(dprep)
```

Read the data file:

```
my.data <- read.csv('dataset14.csv')
```

Perform the test:

```
mardia(cbind(my.data, rep(1, nrow(my.data))))
```

Warning: Mardia's test is compute-intensive and can take a while to complete in R.

Mardia’s Coefficients

```
Mardia's test for class 1
mard1= 172.8751
pvalue for m3= 0
mard2= 0.9525715
p-value for m4= 0.3408072
There is not statistical evidence for normality in
class 1
```

- R reports the test statistics for the coefficients, not the coefficients themselves, i.e. not eqs. 2.23 and 3.12 in (Mardia, 1970), but eqs. 2.26 and 3.20 in (Mardia, 1970).
- R performs a significance test for those test statistics and reports the p-values
  - mard1 is the test statistic for *skewness*
  - mard2 is the test statistic for *kurtosis*
- ML estimates are more sensitive to kurtosis than to skewness

MV Shapiro Test

H_0: The data is multivariate normal (reject H_0 if test is significant)

Load the `mvnormtest` library:

```
library(mvnormtest)
```

Read the data file:

```
my.data <- read.csv(’dataset14.csv’)
```

Perform the test:

```
mshapiro.test(t(my.data))

data:  Z
W = 0.9947, p-value = 8.986e-08
```

Options for Non-Normal Data

- Bootstrapping
  - Bootstrapping of parameter estimates
  - Bootstrapping of χ^2 statistic (”Bollen-Stine-Bootstrap”)
- Adjustment by Satorra & Bentler
  - χ^2 corrections
  - Robust parameter estimates
- Data transformations
  - Probably only feasible for univariate non-normality

Bootstrapping of Model Parameters

```
library(boot)
# Read data and model
data2 <- read.csv(’dataset2.csv’)
model2 <- readLines(’model2.txt’)

# Set up function returning the statistic
my.sem <- function(data, model) {
   # run the model with this data
   r <- sem(model, data=data)
   # return the parameter estimates
   c(r@Fit@x)
}

# Generating a new sample (e.g. by ML)
my.dat.ml <- function(data, mle) {
   as.data.frame(mvrnorm(nrow(data), rep(0, nrow(cov(data))),
      cov(data)))
}
# Generating a new sample (e.g. by resampling)
my.dat.resample <- function(data, mle) {
   sel <- sample.int(nrow(data), nrow(data), replace=TRUE)
   data[sel, ]
}
```

Bootstrapping of Model Parameters (continued)

```
# Performing the bootstrap function
boot.res <- boot(data2, my.sem, 100, sim="parametric",
   ran.gen=my.dat.resample, mle=NULL, model=model2)

# Printing the results
boot.res

# Plotting the results
plot(boot.res, 2)
```

Bootstrap Results

```
PARAMETRIC BOOTSTRAP

Call:
boot(data = data2, statistic = my.sem, R = 100, sim = "parametric",
    ran.gen = my.dat.resample, mle = NULL, model = model2)

Bootstrap Statistics :
       original        bias    std. error
t1*  0.82061034  4.241759e-04 0.0024527151
t2*  0.68356011  3.652468e-04 0.0022986378
t3*  0.71928936 -2.228826e-03 0.0191734676
t4*  0.89811741 -2.044772e-03 0.0196181545
t5*  0.06049773 -3.871053e-05 0.0012860744
t6*  0.02267221 -3.552531e-05 0.0007852346
t7*  0.03988268  1.310166e-04 0.0006400830
t8*  0.24511952 -4.785277e-05 0.0035402181
t9*  0.08918451  2.090452e-04 0.0013230670
t10* 0.03960032  9.083841e-05 0.0009155031
t11* 1.20479992 -2.410636e-04 0.0184002753
t12* 0.07795599  4.960896e-04 0.0033897557
t13* 0.27249081  1.011638e-03 0.0068147467
```

Bootstrap Results

- The standard errors are the square root of the variance of the bootstrap statistics

```
std.err <- sqrt(apply(boot.res$t, 2, var))
```

- The bias is the difference between the mean bootstrap results and the original estimate, i.e.

```
bias <- colMeans(boot.res$t) - boot.res$t0
```

Bootstrap Significance Testing

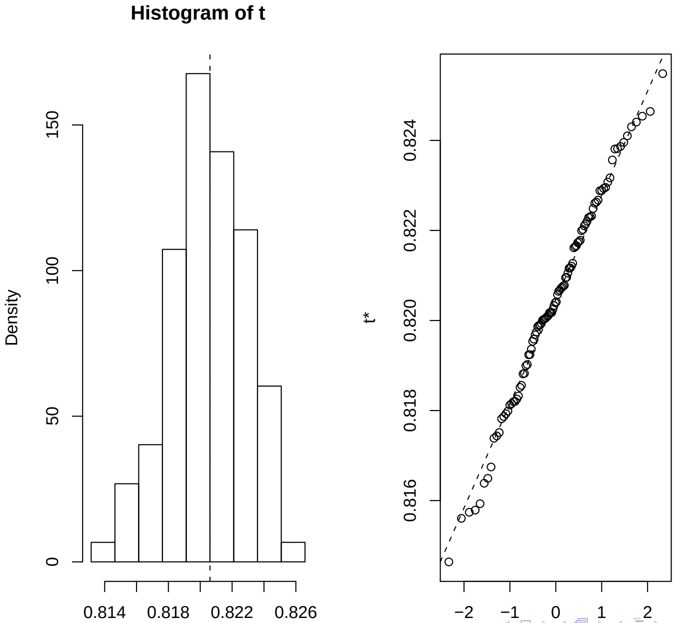
- The original parameter estimate can now be tested for statistical significance (i.e. against 0 or some other value β_0). The following test statistic is normally distributed (for ML estimates $\hat{\beta}$, it is t-distibuted for OLS estimates $\hat{\beta}$):

$$z = \frac{\hat{\beta} - \beta_0}{s.e.(\hat{\beta})}$$

- Compute the z-statistic and test for significance using the normal probability function:

```
z <- boot.res$t0 / std.err
p <- pnorm(z, lower.tail=F)*2
```

Bootstrap Plot



Bootstrapping of χ^2 Statistic (Bollen and Stine, 1993)

```
library(boot)
# Read data and model
data2 <- read.csv(’dataset2.csv’)
model2 <- readLines(’model2.txt’)

# Set up function returning the statistic
my.sem <- function(data, model) {
   # run the model with this data
   r <- sem(model, data=data)
   # Return the fit test statistic
   c(r@Fit@test[[1]]$stat)
}

# Generating a new sample (e.g. by ML)
my.dat.ml <- function(data, mle) {
   as.data.frame(mvrnorm(nrow(data), rep(0, nrow(cov(data))),
                         cov(data)))
}
# Generating a new sample (e.g. by resampling)
my.dat.resample <- function(data, mle) {
   sel <- sample.int(nrow(data), nrow(data), replace=TRUE)
   data[sel, ]
}
```

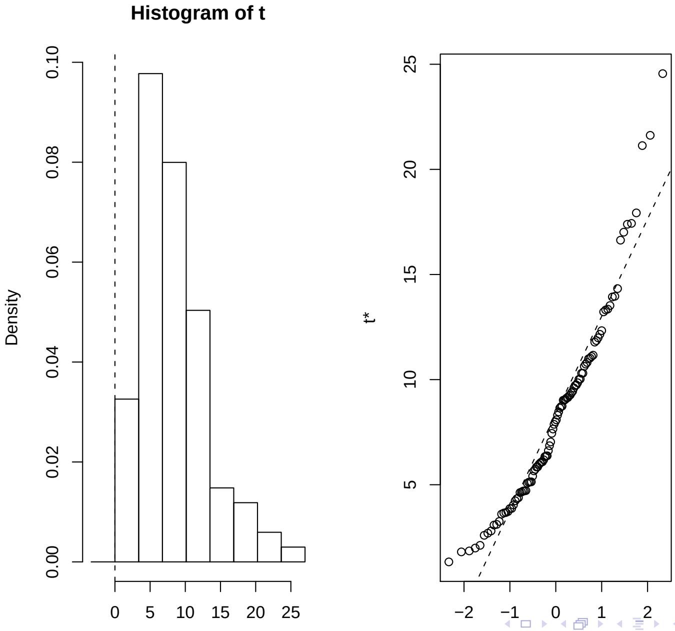
Bootstrapping of χ^2 Statistic (Bollen and Stine, 1993) (continued)

```
# Modify the data sample as per Bollen & Stine (1993)
modified.data <- function(data, model) {
   # zero center the data
   ctr.data <- data - mean(data)
   # run the model for the estimated covariance matrix
   r <- sem(model, data=ctr.data)
   # return adjusted data
   mod.data <- data.frame(as.matrix(ctr.data) %*%
                          solve(chol(cov(ctr.data))) %*%
                          chol(r@Fit@Sigma.hat[[1]]))
   names(mod.data) <- names(data)
   mod.data
}

# Create the modified data
mod.data <- modified.data(data2, model2)
# Performing the bootstrap function
boot.res <- boot(mod.data, my.sem, 100, sim="parametric",
                 ran.gen=my.dat.resample, mle=NULL, model=model2)
# Printing the results
boot.res
# Plotting the results
plot(boot.res)
```

Bollen-Stine Bootstrap Results

```
PARAMETRIC BOOTSTRAP

Call:
boot(data = mod.data, statistic = my.sem, R = 100, sim = "parametric",
    ran.gen = my.dat.resample, mle = NULL, model = model2)

Bootstrap Statistics :
    original   bias    std. error
t1*        0 8.384844    4.626714
```

Bollen-Stine Bootstrap Plot



Bollen-Stine Bootstrap Testing

- Test for model fit with the bootstrapped χ^2 statistic as with the ordinary χ^2 statistic

```
pchisq(8.3848, 8, lower.tail=F)
```

SB Correction and Robust Estimates

Read the model file:

```
model15 <- readLines('model15.txt')
```

Read the data file:

```
data15 <- read.csv('dataset15.csv')
```

Call the `sem` function to estimate the model:

```
sem15 <- sem(model15, data=data15,
estimator='MLM')
```

Print the summary:

```
summary(sem15)
```

```
Lavaan (0.4-9) converged normally after 97 iterations

  Number of observations                           2500

  Estimator                                          ML       Robust
  Minimum Function Chi-square                    15.117       14.829
  Degrees of freedom                                  8            8
  P-value                                         0.057        0.063
  Scaling correction factor                                    1.019
    for the Satorra-Bentler correction (Mplus variant)

Parameter estimates:

  Information                                  Expected
  Standard Errors                            Robust.mlm
```

SB Scaled χ^2 Difference Test

$$
\begin{aligned}
c_d &= \frac{df_0 \times c_0 - df_1 \times c_1}{df_0 - df_1} \\
\Delta\chi^2 &= \frac{\chi^2_0 - \chi^2_1}{c_d}
\end{aligned}
$$

where

$$
\begin{aligned}
df_0, df_1 &= \text{Degrees of freedom of the two models} \\
c_0, c_1 &= \text{SB Scaling correction factors of the two models}
\end{aligned}
$$

Categorical Variables

- For example from 3- or 5-point Likert scales
- Use tetrachoric and polychoric correlations
  - Using the R package `polycor`
  - Good estimates, but biased standard errors and χ^2 (use bootstrapping to correct)
- Use Robust Weighted Least Squares estimator (`estimator='WLS'`)
- Alternatively, use multiple group analysis

Polychoric Correlations

Load the `polycor` package:

```
library(polycor)
```

Read the data file:

```
data16 <- read.csv(’dataset16.csv’)
```

Compute the ”standard” Pearson correlations:

```
cor(data16)
```

Convert to categorical and compute polychoric correlations:

```
cdata <- data.frame(as.factor(data16$a),
    as.factor(data16$b), as.factor(data16$c))
names(cdata) <- c(’a’, ’b’, ’c’)
as.matrix(hetcor(cdata, sdt.err=F, ML=T))
```

Pearson correlations:

```
           a          b         c
a 1.0000000 0.7118257 -0.2500000
b 0.7118257 1.0000000 -0.4460775
c -0.2500000 -0.4460775 1.0000000
```

Polychoric correlations:

```
           a          b         c
a 1.0000000 0.8216439 -0.3247925
b 0.8216439 1.0000000 -0.5252523
c -0.3247925 -0.5252523 1.0000000
```

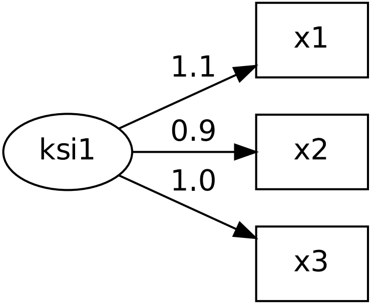
Multi-Group Analysis

Measurement/Factorial Invariance and Interpretational Confounding

- Latents are defined (in part) by their indicators (measurement model)
  - Including loadings, variances, and error variances
- To remain the same latent (in different sample groups, or at different times), it must possess the same indicators, with the same loadings, variances, and error variances
- Measurement/Factorial invariance can and should be imposed and tested in a multiple-group (and/or longitudinal setting
  - Ensure that subjects in each group (at different times) assign the same meaning to the measurement items and thus to the latent variables
- Interpretational confounding occurs when factorial invariance is violated (between groups, or between studies)

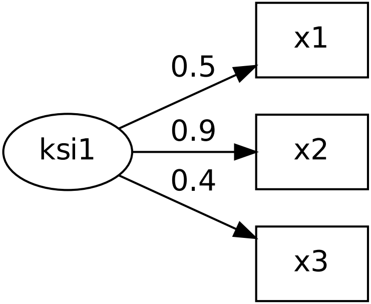
- Factorial invariance is necessary for cumulative knowledge building
- Factorial invariance requires publishing of, and paying attention to, full measurement results (not just loadings)
- Interpretational confound most often occurs with ”formative indicators”

Fictitious Example



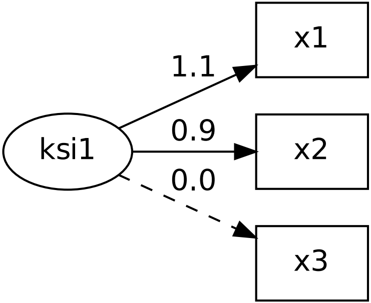
The first model/group/time

Fictitious Example



The second model/group/time: Is this ξ the same ξ as before?

Fictitious Example



How about now?

Progressively test for/constrain:

1. Identical indicators (Configural equivalence)
2. Identical loadings (Metric equivalence)
3. Identical latent covariances (Latent covariance equivalence)
4. Identical measurement errors (Strict factorial equivalence)
5. *Identical observed means (scalar invariance)*

These are nested models, so the χ^2 difference test can be used

Multiple Group Fit Function

For the single-group case:

$$W(\boldsymbol{S}, \boldsymbol{\Sigma}, n) = \frac{e^{-\frac{1}{2} trace(\boldsymbol{S}\boldsymbol{\Sigma}^{-1})}}{|\boldsymbol{\Sigma}|^{n/2}} C$$

For multiple groups:

$$\begin{aligned}
&= W(\boldsymbol{S}_1, \boldsymbol{\Sigma}_1, n_1) W(\boldsymbol{S}_2, \boldsymbol{\Sigma}_2, n_2) W(\boldsymbol{S}_3, \boldsymbol{\Sigma}_3, n_3) \ldots \\
&= \frac{e^{-\frac{1}{2} trace(\boldsymbol{S_1}\boldsymbol{\Sigma_1}^{-1})}}{|\boldsymbol{\Sigma_1}|^{n_1/2}} C_1 \frac{e^{-\frac{1}{2} trace(\boldsymbol{S_2}\boldsymbol{\Sigma_2}^{-1})}}{|\boldsymbol{\Sigma}|^{n_2/2}} C_2 \ldots \\
&= \prod_{g=1}^{G} \left[\frac{e^{-\frac{1}{2} trace(\boldsymbol{S_g}\boldsymbol{\Sigma_g}^{-1})}}{|\boldsymbol{\Sigma_g}|^{n_g/2}} C_g \right]
\end{aligned}$$

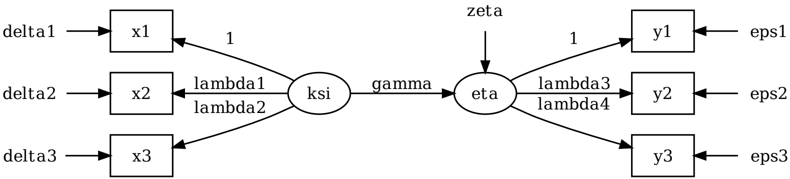
We maximize the log of this likelihood function:

$$
\begin{aligned}
LL &= log\left(\prod_{g=1}^{G}\left[\frac{e^{-\frac{1}{2}trace(\boldsymbol{S_g}\boldsymbol{\Sigma_g}^{-1})}}{|\boldsymbol{\Sigma_g}|^{n_g/2}}C_g\right]\right)\\
&= \sum_{g=1}^{G}\left[-\frac{1}{2}trace(\boldsymbol{S}_g\boldsymbol{\Sigma_g}^{-1}) - \frac{n_g}{2}trace|\boldsymbol{\Sigma_g}| + C_g\right]\\
&= -\frac{1}{2}\sum_{g=1}^{G}n_g\left[log|\boldsymbol{\Sigma}_g| + trace(\boldsymbol{S}_g\boldsymbol{\Sigma}_g^{-1}) + C_g\right]
\end{aligned}
$$

Weighted sum of per-group LL

Per-group $\boldsymbol{\theta}$ may be constrained across groups

Example



Question: Is γ identical between two groups?

Estimation in R

Read the model file:

```
model17 <- readLines('model17.txt')
```

Read the data file:

```
data17 <- read.csv('dataset17.csv')
```

Call the `sem` function to estimate the model:

```
sem17.base <- sem(model17, data=data17,
group='Group')
```

Print the summary:

```
summary(sem17.base)
```

Alternatively, from covariance matrices

Two data sets:

```
data17.1 <- data17[data17$Group==’1’,1:6]
data17.2 <- data17[data17$Group==’2’,1:6]
```

Call the `sem` function to estimate the model:

```
sem17.base <- sem(model17,
 sample.cov=list(cov(data17.1), cov(data17.2)),
 sample.nobs=list(nrow(data17.1), nrow(data17.2)))
```

Print the summary:

```
summary(sem17.base)
```

```
Lavaan (0.4-9) converged normally after 185 iterations

  Number of observations per group
  1                                                 1000
  2                                                 1000

  Estimator                                           ML
  Minimum Function Chi-square                     10.832
  Degrees of freedom                                  16
  P-value                                          0.820

Chi-square for each group:

  1                                                7.614
  2                                                3.218
```

Metric Equivalence

Call the `sem` function to estimate the model:

```
sem17.metric <- sem(model17, data=data17,
   group='Group',
   group.equal='loadings')
```

Print the summary:

```
summary(sem17.metric)
```

```
Lavaan (0.4-9) converged normally after 169 iterations

  Estimator                                         ML
  Minimum Function Chi-square                   12.611
  Degrees of freedom                                20
  P-value                                        0.893
```

Compute $\Delta\chi^2$:

```
delta.chisq <- 12.611 - 10.832
```

Compute Δdf:

```
delta.df <- 20 - 16
```

Calculate the p-value:

```
p.value <- pchisq(delta.chisq, delta.df, lower.tail=F)
```

And print it:

```
p.value
[1] 0.7763221
```

Latent Covariance Equivalence

Read the model file:

```
model17 <- readLines('model17a.txt')
```

```
ksi =~ NA*x1 + x2 + x3
eta =~ y1 + y2 + y3
eta ~ ksi
ksi ~~ c(1, 1)*ksi
```

Call the `sem` function to estimate the model:

```
sem17.latent.cov <- sem(model17, data=data17,
    group='Group',
    group.equal='loadings')
```

Print the summary:

```
summary(sem17.latent.cov)
```

```
Lavaan (0.4-9) converged normally after 224 iterations

  Estimator                                         ML
  Minimum Function Chi-square                   12.619
  Degrees of freedom                                21
  P-value                                        0.921
```

Compute $\Delta\chi^2$:

```
delta.chisq <- 12.619 - 12.611
```

Compute Δdf:

```
delta.df <- 21 - 20
```

Calculate the p-value:

```
p.value <- pchisq(delta.chisq, delta.df, lower.tail=F)
```

And print it:

```
p.value
[1] 0.7772974
```

Strict Factorial Equivalence

Call the `sem` function to estimate the model:

```
sem17.strict <- sem(model17, data=data17,
 group='Group',
 group.equal=c('loadings','residuals'))
```

Print the summary:

```
summary(sem17.strict)
```

```
Lavaan (0.4-9) converged normally after 134 iterations

  Number of observations per group
  1                                                 1000
  2                                                 1000

  Estimator                                           ML
  Minimum Function Chi-square                     28.281
  Degrees of freedom                                  27
  P-value                                          0.397
```

Compute $\Delta\chi^2$:

```
delta.chisq <- 28.281 - 12.619
```

Compute Δdf:

```
delta.df <- 27 - 21
```

Calculate the p-value:

```
p.value <- pchisq(delta.chisq, delta.df, lower.tail=F)
```

And print it:

```
p.value
[1] 0.01568784
```

Testing γ

Read the model file:

```
model17 <- readLines(’model17b.txt’)
```

```
ksi =~ NA*x1 + x2 + x3
eta =~ y1 + y2 + y3
eta ~ label(c(’gamma’, ’gamma’))*ksi
ksi ~~ c(1, 1)*ksi
```

Call the `sem` function to estimate the model:

```
sem17.gamma <- sem(model17, data=data17,
 group=’Group’,
 group.equal=c(’loadings’,’residuals’))
```

Print the summary:

```
summary(sem17.gamma)
```

```
Lavaan (0.4-9) converged normally after 148 iterations

  Number of observations per group
  1                                                 1000
  2                                                 1000

  Estimator                                           ML
  Minimum Function Chi-square                     59.135
  Degrees of freedom                                  28
  P-value                                          0.001
```

Compute $\Delta\chi^2$:

```
delta.chisq <- 59.135 - 28.281
```

Compute Δdf:

```
delta.df <- 28-27
```

Calculate the p-value:

```
p.value <- pchisq(delta.chisq, delta.df, lower.tail=F)
```

And print it:

```
p.value
[1] 2.781881e-08
```

Simple, Pre-defined Function

```
model17 <- readLines(’model17.txt’)

measurementInvariance(
   model17, data=data17, group=’Group’)
```

```
Measurement invariance tests:

Model 1: configural invariance:
    chisq        df    pvalue       cfi     rmsea       bic
   10.832    16.000     0.820     1.000     0.000 -2873.743

Model 2: weak invariance (equal loadings):
    chisq        df    pvalue       cfi     rmsea       bic
   12.611    20.000     0.893     1.000     0.000 -2902.368

 [Model 1 versus model 2]
  delta.chisq      delta.df delta.p.value     delta.cfi
        1.779         4.000         0.776         0.000

Model 3: strong invariance (equal loadings + intercepts):
    chisq        df    pvalue       cfi     rmsea       bic
   16.113    24.000     0.884     1.000     0.000 -2838.058

 [Model 1 versus model 3]
  delta.chisq      delta.df delta.p.value     delta.cfi
        5.281         8.000         0.727         0.000

 [Model 2 versus model 3]
  delta.chisq      delta.df delta.p.value     delta.cfi
        3.502         4.000         0.478         0.000

Model 4: equal loadings + intercepts + means:
    chisq        df    pvalue       cfi     rmsea       bic
   22.941    26.000     0.636     1.000     0.000 -2846.432

 [Model 1 versus model 4]
  delta.chisq      delta.df delta.p.value     delta.cfi
       12.109        10.000         0.278         0.000

 [Model 3 versus model 4]
  delta.chisq      delta.df delta.p.value     delta.cfi
        6.828         2.000         0.033         0.000
```

Missing Values

”Missingness”

- MAR: Missing at random
  - The ”missingness” of missing values depends on values of other variables
  - E.g. for socially or personally sensitive questions
- MCAR: Missing completely at random
  - The ”missingness” of missing values does not depend on values of other variables

Easy but Wrong

Compute covariances using either:

- Listwise deletion
  - Biased parameter estimates under MAR
  - Biased standard errors under MAR
  - Reduced sample size → reduced power
- Pairwise deletion
  - Biased parameter estimates under MAR
  - Biased standard errors under MAR
  - Unclear sample size

Missing Values in SEM

- Multiple group approach
  - Each pattern of "missingness" is its own group with its own group sample size and group covariance matrix
  - Manually only feasible for small number of "missingness" patterns
- Direct ML / Raw ML / FIML
  - The limiting case of the multiple group approach, each sample point is its own group
- Imputation of Data (e.g. EM, MI, etc.)

FIML

Maximize the log-likelihood function

$$LL(\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{i=1}^{N} \left[K_i - \frac{1}{2} log|\boldsymbol{\Sigma_i}| - \frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1} (\boldsymbol{x}_i - \boldsymbol{\mu}_i) \right]$$

Does not yield a model χ^2 statistic but a *data* χ^2 statistic that can be used for a $\Delta\chi^2$ test to an unconstrained base model.

FIML in R

Read the model file:

```
model18 <- readLines(’model18.txt’)
```

Read the dataset:

```
data18 <- read.csv(’dataset18.csv’)
```

Call the `sem` function to estimate the model:

```
sem18 <- sem(model18, data=data18,
missing=’FIML’)
```

Print the summary:

```
summary(sem18)
```

```
Lavaan (0.4-9) converged normally after 54 iterations

  Number of observations                           301

  Number of missing patterns                        36

  Estimator                                         ML
  Minimum Function Chi-square                   82.631
  Degrees of freedom                                24
  P-value                                        0.000
```

EM Imputation (norm)

Load the `norm` library:

```
library(norm)
```

Preliminary processing of data:

```
s <- prelim.norm(data.matrix(data18))
```

Parameter estimation using EM algorithm:

```
thetahat <- em.norm(s)
```

Extract $\hat{\Sigma}$ from the estimated parameters:

```
data18.cov <- getparam.norm(s, thetahat, corr=FALSE)$sigma
dimnames(data18.cov)[[1]] <- names(data18)
dimnames(data18.cov)[[2]] <- names(data18)
```

Do the SEM estimation:

```
sem18.norm <- sem(model18,
    sample.cov=data18.cov,
    sample.nobs=nrow(data18))
```

Print the summary:

```
summary(sem18.norm)
```

```
Lavaan (0.4-9) converged normally after 36 iterations

  Number of observations                           301

  Estimator                                         ML
  Minimum Function Chi-square                   98.231
  Degrees of freedom                                24
  P-value                                        0.000
```

EM Imputation (mvnmle)

Load the `mvnmle` library:

```
library(mvnmle)
```

Estimate $\hat{\Sigma}$:

```
data18.cov <- mlest(data18)$sigmahat
dimnames(data18.cov)[[1]] <- names(data18)
dimnames(data18.cov)[[2]] <- names(data18)
```

Do the SEM estimation:

```
sem18.mvnmle <- sem(model18,
     sample.cov=data18.cov,
     sample.nobs=nrow(data18))
```

Print the summary:

```
summary(sem18.mvnmle)
```

```
Lavaan (0.4-9) converged normally after 36 iterations

  Number of observations                           301

  Estimator                                         ML
  Minimum Function Chi-square                   98.235
  Degrees of freedom                                24
  P-value                                        0.000
```

MI (norm)

Load the `norm` package:

```
library(norm)
```

Initialize random number generator and results list:

```
cdata <- vector("list", 1)
my.sem <- vector("list", 1)
rngseed(987654321)
```

Preliminary processing of data:

```
s <- prelim.norm(data.matrix(data18))
```

Parameter imputation using EM algorithm:

```
thetahat <- em.norm(s)
```

Generate 100 samples and estimate SEM for each:

```
for(i in 1:100) {
  cdata[[i]] <- imp.norm(s, thetahat, data18)
  my.sem[[i]] <- sem(model18, data=cdata[[i]])
}
```

MI (norm), continued

Initialize estimates and stderrs:

```
est <- vector("list", 1)
se <- vector("list", 1)
```

Collect estimates and stderrs from 100 SEM runs and add names to results:

```
for(i in 1:100) {
 e <- parameterEstimates(my.sem[[i]])
 est[[i]] <- e$est
 se[[i]] <- e$se
 names(est[[i]])<-paste(e$lhs,e$op,e$rhs,sep=”)
 names(se[[i]])<-paste(e$lhs,e$op,e$rhs,sep=”)
}
```

Perform MI inference:

```
mi.inf <- mi.inference(est, se)
```

```
$est
   f1=~x1    f1=~x2    f1=~x3    f2=~x4    f2=~x5    f2=~x6    f3=~x7    f3=~x8
1.0000000 0.5861326 0.7777703 1.0000000 1.0912204 0.9059428 1.0000000 1.2227900
   f3=~x9    x1~~x1    x2~~x2    x3~~x3    x4~~x4    x5~~x5    x6~~x6    x7~~x7
1.1951371 0.6270894 1.1162524 0.8073750 0.3667953 0.4616144 0.3485899 0.8381147
   x8~~x8    x9~~x9    f1~~f1    f2~~f2    f3~~f3    f1~~f2    f1~~f3    f2~~f3
0.5091148 0.5285973 0.7726876 0.9922892 0.3422500 0.4156877 0.2648518 0.1560980

$std.err
    f1=~x1     f1=~x2     f1=~x3     f2=~x4     f2=~x5     f2=~x6     f3=~x7
0.00000000 0.10675158 0.11959593 0.00000000 0.06732787 0.05698209 0.00000000
    f3=~x8     f3=~x9     x1~~x1     x2~~x2     x3~~x3     x4~~x4     x5~~x5
0.18377574 0.22049125 0.11323679 0.10429933 0.09547801 0.05088298 0.06186385
    x6~~x6     x7~~x7     x8~~x8     x9~~x9     f1~~f1     f2~~f2     f3~~f3
0.04421830 0.09120055 0.08960748 0.08801903 0.14444652 0.11511241 0.09119267
    f1~~f2     f1~~f3     f2~~f3
0.07721910 0.05718760 0.04768741

$signif
      f1=~x1       f1=~x2       f1=~x3       f2=~x4       f2=~x5       f2=~x6
         NaN 4.111662e-08 8.756662e-11          NaN 0.000000e+00 0.000000e+00
      f3=~x7       f3=~x8       f3=~x9       x1~~x1       x2~~x2       x3~~x3
         NaN 2.900968e-11 7.915268e-08 3.135554e-08 0.000000e+00 0.000000e+00
      x4~~x4       x5~~x5       x6~~x6       x7~~x7       x8~~x8       x9~~x9
6.061818e-13 9.192647e-14 3.330669e-15 0.000000e+00 1.776217e-08 2.729077e-09
      f1~~f1       f2~~f2       f3~~f3       f1~~f2       f1~~f3       f2~~f3
8.931747e-08 0.000000e+00 1.787087e-04 7.417008e-08 3.647626e-06 1.063100e-03
```

MI (Amelia)

Load the `Amelia` library:

```
library(Amelia)
```

Prepare data structures for results:

```
my.sem <- vector("list", 1)
```

Generate 100 imputed samples:

```
cdata <- amelia(data18, 100, 0)
```

Look at the imputed data:

```
summary(cdata)
plot(cdata)
```

Estimate the SEM for each imputed data set:

```
for(i in 1:100) {
  my.sem[[i]]<-sem(model18,
     data=cdata$imputations[[i]]) }
```

MI (Amelia), continued

Initialize estimates and stderrs:

```
est <- vector("list", 1)
se <- vector("list", 1)
```

Collect estimates and stderrs from 100 SEM runs and add names to results:

```
for(i in 1:100) {
 e <- parameterEstimates(my.sem[[i]])
 est[[i]] <- e$est
 ste[[i]] <- e$se
 names(est[[i]])<-paste(e$lhs, e$op, e$rhs, sep=”)
 names(se[[i]])<-paste(e$lhs, e$op, e$rhs, sep=”)
}
```

Perform MI inference:

```
mi.inf <- mi.inference(est, se)
```

```
$est
   f1=~x1    f1=~x2    f1=~x3    f2=~x4    f2=~x5    f2=~x6    f3=~x7    f3=~x8 
1.0000000 0.5818409 0.7814281 1.0000000 1.0947343 0.9090930 1.0000000 1.2198461 
   f3=~x9    x1~~x1    x2~~x2    x3~~x3    x4~~x4    x5~~x5    x6~~x6    x7~~x7 
1.1854478 0.6268453 1.1213662 0.8107859 0.3699171 0.4588627 0.3490364 0.8341486 
   x8~~x8    x9~~x9    f1~~f1    f2~~f2    f3~~f3    f1~~f2    f1~~f3    f2~~f3 
0.5061382 0.5354579 0.7725229 0.9898463 0.3465683 0.4168375 0.2657627 0.1551272 

$std.err
    f1=~x1     f1=~x2     f1=~x3     f2=~x4     f2=~x5     f2=~x6     f3=~x7 
0.00000000 0.10770520 0.12033975 0.00000000 0.06831416 0.05839695 0.00000000 
    f3=~x8     f3=~x9     x1~~x1     x2~~x2     x3~~x3     x4~~x4     x5~~x5 
0.18525491 0.21553616 0.11587055 0.10625407 0.09540465 0.05215444 0.06189921 
    x6~~x6     x7~~x7     x8~~x8     x9~~x9     f1~~f1     f2~~f2     f3~~f3 
0.04517067 0.09062315 0.08632220 0.08596152 0.14692484 0.11666523 0.09317247 
    f1~~f2     f1~~f3     f2~~f3 
0.07782708 0.05788680 0.04799656 

$signif
      f1=~x1       f1=~x2       f1=~x3       f2=~x4       f2=~x5       f2=~x6 
         NaN 6.802586e-08 9.360401e-11          NaN 0.000000e+00 0.000000e+00 
      f3=~x7       f3=~x8       f3=~x9       x1~~x1       x2~~x2       x3~~x3 
         NaN 4.699885e-11 5.043186e-08 6.587479e-08 0.000000e+00 0.000000e+00 
      x4~~x4       x5~~x5       x6~~x6       x7~~x7       x8~~x8       x9~~x9 
1.497025e-12 1.332268e-13 1.265654e-14 0.000000e+00 5.680667e-09 6.540031e-10 
      f1~~f1       f2~~f2       f3~~f3       f1~~f2       f1~~f3       f2~~f3 
1.489414e-07 0.000000e+00 2.050689e-04 8.673901e-08 4.437665e-06 1.229649e-03 
```

Coefficient Estimates

	FIML	EM	MI (norm)	MI (Amelia)	Pairwise	Listwise
x_2	.566	.591	.586	.581	.563	.510
x_3	.774	.783	.777	.781	.760	.710
x_5	1.087	1.093	1.091	1.095	1.070	.992
x_6	.905	.905	.906	.909	.894	.883
x_8	1.210	1.215	1.223	1.220	1.272	1.055
x_9	1.246	1.180	1.195	1.185	1.434	1.316
f_1	.784	.765	.773	.773	.788	.926
f_2	.996	.987	.992	.990	.993	1.073
f_3	.329	.343	.342	.347	.283	.363

Note: EM estimates for `norm` and `mvnmle` are identical.

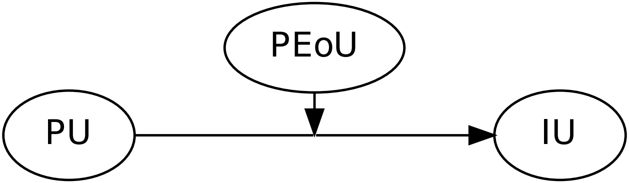
Standard Error Estimates

	FIML	EM	MI (norm)	MI (Amelia)	Pairwise	Listwise
x_2	.116	.102	.106	.108	.097	.084
x_3	.129	.111	.120	.120	.104	.090
x_5	.069	.065	.067	.068	.066	.060
x_6	.059	.055	.057	.058	.055	.054
x_8	.173	.176	.184	.185	.193	.149
x_9	.265	.171	.220	.216	.217	.177
f_1	.152	.138	.144	.147	.138	.141
f_2	.117	.113	.115	.117	.114	.118
f_3	.097	.082	.091	.093	.074	.082

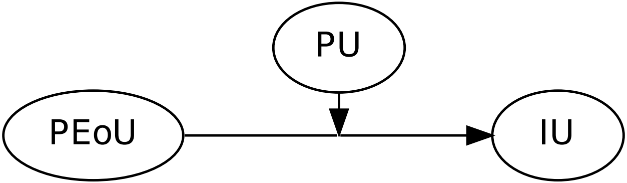
Note: EM estimates for `norm` and `mvnmle` are identical.

- Both listwise and pairwise deletion lead to unpredictably biased parameter estimates
- Both listwise and pairwise deletion lead to wrong standard errors
  - Significance of parameters may be over- or under-estimated
- FIML produces results close to MI and single EM
- MI and single EM do not differ much

Interaction/Moderation

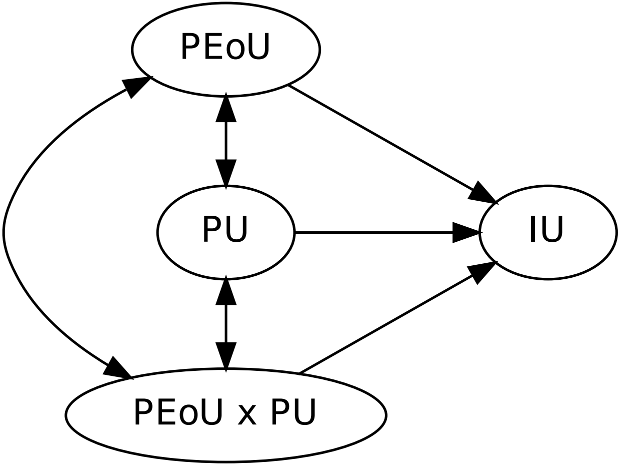


Values of PEoU affect the relationship of PU on IU



Values of PU affect the relationship of PEoU on IU

Equivalent to previous model



Model explicitly includes main and interaction effects

Interaction in SEM

- Usually only two-latent interaction (but more possible)
- Interaction terms violate multivariate normality
  - ML estimates are robust if errors and latents are uncorrelated
  - Use SB robust estimates or any other technique for non-normal data
- Indicators of interactions are products of indicators of main effects
  - They are complex functions of main-effects
  - Estimating variance and covariance is problematic
- Interaction latent variables do not typically represent theoretical constructs
- Errors of interaction indicators are likely related to errors of main effect indicators
  - Possible multi-collinearity issues
  - When modelling all such relationships, the model may become unidentified

Cortina, Chen, Dunlap (2001), ORM (4) 4.

Assume four indicators of two exogenous latents:

$$
\begin{aligned}
x_1 &= \xi_1 + \epsilon_1 \\
x_2 &= \lambda_{21}\xi_1 + \epsilon_2 \\
x_3 &= \xi_2 + \epsilon_3 \\
x_4 &= \lambda_{42}\xi_2 + \epsilon_4
\end{aligned}
$$

Four possible product indicators are then:

$$
\begin{aligned}
x_5 &= x_1 x_3 = \xi_1\xi_2 + \xi_1\epsilon_3 + \xi_2\epsilon_1 + \epsilon_1\epsilon_3 \\
x_6 &= x_1 x_4 = \lambda_{42}\xi_1\xi_2 + \xi_1\epsilon_4 + \lambda_{42}\xi_2\epsilon_1 + \epsilon_1\epsilon_4 \\
x_7 &= x_2 x_3 = \lambda_{21}\xi_1\xi_2 + \lambda_{21}\xi_1\epsilon_3 + \xi_2\epsilon_2 + \epsilon_2\epsilon_3 \\
x_8 &= x_2 x_4 = \lambda_{21}\lambda_{42}\xi_1\xi_2 + \lambda_{21}\xi_1\epsilon_4 + \lambda_{42}\xi_2\epsilon_2 + \epsilon_2\epsilon_4
\end{aligned}
$$

Multiple Approaches

- Jaccard & Wan (1995): All product indicators
- Jörsekog & Yang (1996): Single product indicator
- Ping (1995): Single product indicator, two step process
- Matheiu et al. (1992): Single product indicator
- Ping (1996): Single product indicator, two step process

Ping (1995)

Single product indicator

$$x_{\times} = (x_1 + x_2)(x_3 + x_4)$$

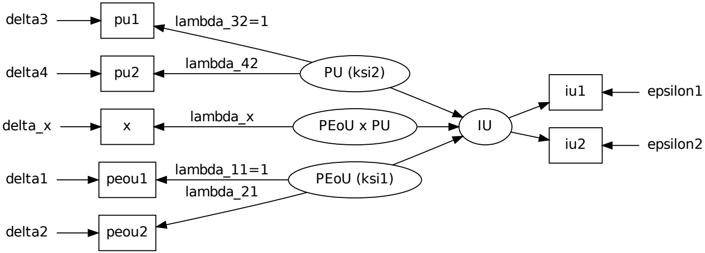
With loading

$$\lambda_{x_{\times}} = (\lambda_{11} + \lambda_{21})(\lambda_{32} + \lambda_{42})$$

With error variance

$$\begin{aligned}\theta_{x_{\times}} &= (\lambda_{11} + \lambda_{21})^2 Var(\xi_1)(\theta_{x_3} + \theta_{x_4}) \\ &\quad + (\lambda_{32} + \lambda_{42})^2 Var(\xi_2)(\theta_{x_1} + \theta_{x_1}) \\ &\quad + (\theta_{x_3} + \theta_{x_4})(\theta_{x_1} + \theta_{x_1})\end{aligned}$$

Example



Note that $\lambda_{11} = \lambda_{32} = 1$.

First Step Estimation in R

Read the model file:

```
model19.1 <- readLines(’model19step1.txt’)
```

Read the dataset:

```
data19 <- read.csv(’dataset19.csv’)
```

Call the `sem` function to estimate the model:

```
sem19.1 <- sem(model19.1, data=data19)
```

Print the summary:

```
summary(sem19.1)
```

```
Lavaan (0.4-9) converged normally after 86 iterations

  Number of observations                           2500

  Estimator                                          ML
  Minimum Function Chi-square                     7.599
  Degrees of freedom                                  6
  P-value                                         0.269

                   Estimate  Std.err  Z-value  P(>|z|)
Latent variables:
  pu =~
    pu2               1.219    0.006  189.051    0.000
  peou =~
    peou2             0.822    0.004  189.693    0.000

Variances:
    pu1               0.024    0.003    8.691    0.000
    pu2               0.020    0.004    4.933    0.000
    peou1             0.022    0.004    5.753    0.000
    peou2             0.021    0.003    7.833    0.000
    pu                0.830    0.024   34.174    0.000
    peou              1.203    0.035   34.501    0.000
```

Compute Path Coefficient and Error Variance

$$\begin{aligned} \lambda_{x_\times} &= (1+1.219)(1+.822) \\ &= 4.043 \end{aligned}$$

With error variance

$$\begin{aligned} \theta_{x_\times} &= (1+1.219)^2 \times .830 \times (.022+.021) \\ &\quad + (1+.822)^2 \times 1.203 \times (.024+.020) \\ &\quad + (.022+.021)(.024+0.20) \\ &= 4.923961 \times .830 \times .043 \\ &\quad + 3.319684 \times 1.203 \times .044 \\ &\quad + .001892 \\ &= .3533 \end{aligned}$$

Specify Step 2 Model

```
iu =~ iu1 + iu2
pu =~ pu1 + pu2
peou =~ peou1 + peou2
iu ~ pu + peou + cross
cross =~ 4.043*cross1
cross1 ~~ 0.3533*cross1
```

Second Step Estimation in R

Construct the interaction indicator:

```
cross1 <- (data19$pu1+data19$pu2)
   * (data19$peou1+data19$peou2)
data19new <- cbind(data19, cross1)
```

Read the model file:

```
model19.2 <- readLines(’model19step2.txt’)
```

Call the `sem` function to estimate the model:

```
sem19.2 <- sem(model19.2, data=data19new)
```

Print the summary:

```
summary(sem19.2)
```

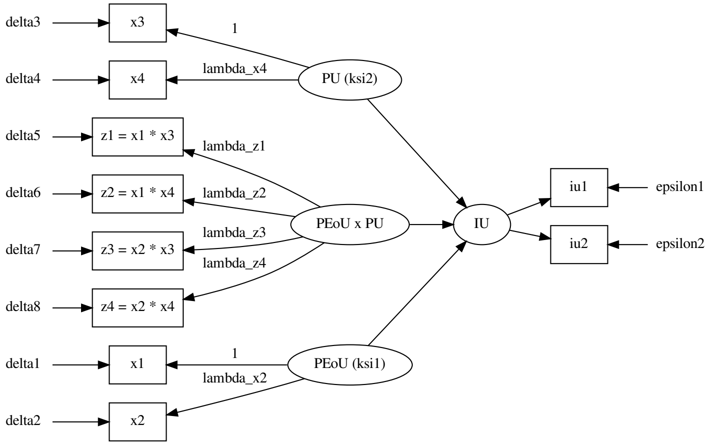
```
Lavaan (0.4-9) converged normally after 108 iterations

  Number of observations                          2500

  Estimator                                         ML
  Minimum Function Chi-square                    8.383
  Degrees of freedom                                 9
  P-value                                        0.496

                   Estimate  Std.err  Z-value  P(>|z|)
Latent variables:
  iu =~
    iu1               1.000
    iu2               1.107    0.006  194.011    0.000
  pu =~
    pu1               1.000
    pu2               1.218    0.005  228.748    0.000
  peou =~
    peou1             1.000
    peou2             0.822    0.004  229.393    0.000
  cross =~
    cross1            4.043
```

```
Regressions:
  iu ~
    pu                0.554    0.004  143.828    0.000
    peou              0.453    0.003  148.160    0.000
    cross             0.334    0.003  112.730    0.000

Covariances:
  pu ~~
    peou              0.018    0.020    0.873    0.383
    cross            -0.051    0.019   -2.682    0.007
  peou ~~
    cross            -0.040    0.023   -1.740    0.082
```

```
Variances:
    cross1            0.353
    iu1               0.023    0.001   26.463    0.000
    iu2               0.021    0.001   22.691    0.000
    pu1               0.023    0.001   19.650    0.000
    pu2               0.020    0.002   12.973    0.000
    peou1             0.022    0.002   13.967    0.000
    peou2             0.021    0.001   18.460    0.000
    iu                0.003    0.001    4.461    0.000
    pu                0.831    0.024   34.373    0.000
    peou              1.203    0.035   34.695    0.000
    cross             1.068    0.031   34.654    0.000
```

Jaccard & Wan (1995)

Multiple product indicators



Jaccard & Wan (1995)

Multiple product indicators

$$\begin{aligned} z_1 &= x_1 * x_3 \\ z_2 &= x_1 * x_4 \\ z_3 &= x_2 * x_3 \\ z_4 &= x_2 * x_4 \end{aligned}$$

With loadings

$$\begin{aligned} \lambda_{z_1} &= 1 \\ \lambda_{z_2} &= \lambda_{x_4} \\ \lambda_{z_3} &= \lambda_{x_2} \\ \lambda_{z_4} &= \lambda_{x_2}\lambda_{x_4} \end{aligned}$$

Latent variances

$$\phi_{33} = \phi_{11}\phi_{22} + \phi_{12}^2$$

Jaccard & Wan (1995)

Error variances

$$
\begin{aligned}
\theta_{55} &= \phi_{11}\theta_{33} + \phi_{22}\theta_{11} + \theta_{11}\theta_{33} \\
\theta_{66} &= \phi_{11}\theta_{44} + \lambda_{x_4}^2\phi_{22}\theta_{11} + \theta_{11}\theta_{44} \\
\theta_{77} &= \lambda_{x_2}^2\phi_{11}\theta_{33} + \phi_{22}\theta_{22} + \theta_{22}\theta_{33} \\
\theta_{88} &= \lambda_{x_2}^2\phi_{11}\theta_{44} + \lambda_{x_4}^2\phi_{22}\theta_{22} + \theta_{22}\theta_{44}
\end{aligned}
$$

Jaccard & Wan (1995)

Read the model file:

```
model19.jw <- readLines(’model19_jw.txt’)
```

Read the dataset:

```
data19 <- read.csv(’dataset19.csv’)
```

Construct the interaction indicator:

```
cross1 <- data19$peou1*data19$pu1
cross2 <- data19$peou1*data19$pu2
cross3 <- data19$peou2*data19$pu1
cross4 <- data19$peou2*data19$pu2
data19new <- cbind(data19, cross1, cross2,
cross3, cross4)
```

Call the `sem` function to estimate the model:

```
sem19.jw <- sem(model19.jw, data=data19new)
```

```
#
# Interaction as per Jaccard and Wen (1995)
#
# From Li et al. in MBR 33(1), 1-39, 1998
#
# cross1 <- peou1*pu1
# cross2 <- peou1*pu2
# cross3 <- peou2*pu1
# cross4 <- peou2*pu2
#
# Define standard reflective latents and label the loading and error
# variance params
#
iu =~ iu1 + iu2
#
peou =~ peou1 + l1*peou2
peou1 ~~ t1*peou1
peou2 ~~ t2*peou2
```

```
pu =~ pu1 + l2*pu2
pu1 ~~ t3*pu1
pu2 ~~ t4*pu2
#
# Interaction term
#
cross =~ cross1 + l3*cross2 + l4*cross3 + l5*cross4
cross1 ~~ t5*cross1
cross2 ~~ t6*cross2
cross3 ~~ t7*cross3
cross4 ~~ t8*cross4
cross1 ~~ cross2
cross1 ~~ cross3
# cross1 ~~ cross4
# cross2 ~~ cross3
cross2 ~~ cross4
cross3 ~~ cross4
```

```
#
# Setup covariances of exogenous latents and label them
#
peou ~~ p1*peou
pu ~~ p2*pu
peou ~~ p12*pu
peou ~~ 0*cross
pu ~~ 0*cross
cross ~~ p3*cross
#
# Model additional constraints
#
# Loadings
#
l3 == l2
l4 == l1
l5 == l1*l2
#
# Latent variances
#
p3 == p1*p2+p12*p12
```

```
# Error variances
#
t5 == p1*t3 + p2*t1 + t1*t3
t6 == p1*t4 + l2*l2*p2*t1 + t1*t4
t7 == l1*l1*p1*t3 + p2*t2 + t2*t3
t8 == l1*l1*p1*t4 + l2*l2*p2*t2 + t2*t4
#
# Regressions
#
iu ~ peou + pu + cross
```

```
Lavaan (0.4-9) converged normally after 994 iterations

  Number of observations                          2500

  Estimator                                         ML
  Minimum Function Chi-square                   32.547
  Degrees of freedom                                30
  P-value                                        0.343

Parameter estimates:

  Information                                 Expected
  Standard Errors                             Standard
```

```
                   Estimate  Std.err  Z-value  P(>|z|)
Latent variables:
  iu =~
    iu1               1.000
    iu2               1.107    0.006  200.437    0.000
  peou =~
    peou1             1.000
    peou2             0.821    0.003  322.148    0.000
  pu =~
    pu1               1.000
    pu2               1.213    0.004  325.989    0.000
  cross =~
    cross1            1.000
    cross2            1.213    0.004  325.989    0.000
    cross3            0.821    0.003  322.148    0.000
    cross4            0.996    0.004  228.929    0.000

Regressions:
  iu ~
    peou              0.453    0.003  153.875    0.000
    pu                0.552    0.004  152.457    0.000
    cross             0.333    0.003  109.134    0.000
```

```
Covariances:
  cross1 ~~
    cross2            0.023    0.001   15.954    0.000
    cross3            0.023    0.001   23.445    0.000
  cross2 ~~
    cross4            0.019    0.001   14.165    0.000
  cross3 ~~
    cross4            0.023    0.001   23.068    0.000
  peou ~~
    pu                0.019    0.021    0.905    0.366
    cross             0.000
  pu ~~
    cross             0.000
```

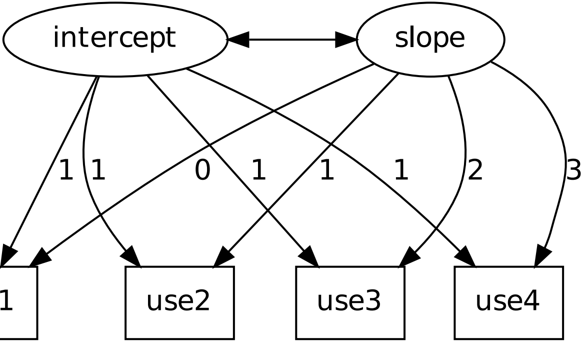
```
Variances:
    peou1             0.021    0.001   15.859    0.000
    peou2             0.022    0.001   22.579    0.000
    pu1               0.024    0.001   23.564    0.000
    pu2               0.020    0.001   15.335    0.000
    cross1            0.047    0.002   29.080    0.000
    cross2            0.052    0.002   22.908    0.000
    cross3            0.039    0.001   33.953    0.000
    cross4            0.045    0.002   27.994    0.000
    peou              1.231    0.028   43.438    0.000
    pu                0.855    0.020   43.045    0.000
    cross             1.053    0.023   45.061    0.000
    iu1               0.023    0.001   26.490    0.000
    iu2               0.021    0.001   22.722    0.000
    iu                0.003    0.001    4.375    0.000
```

Latent Growth Models

Latent Growth Models

- Longitudinal data, treated as time series
- In contrast to ”traditional” (time-lagged) longitudinal models, a LGC does not ”explain”, but only describes the observed growth curve
- Growth parameters, Intercept and Slope, modeled as latent variables
- Growth parameters may be affected by other variables
- Multiple growth curves in a single model possible
- Growth curve explain observations or explains latent variables
  - In the latter case, constraints should ensure factorial invariance
- LGMs include a means model to estimate mean intercept and slope

Simple Example



Describing (*not explaining*) levels of tool use at 4 points in time

Estimation in R

Read the model file:

```
model20 <- readLines('model20.txt')
```

Read the dataset:

```
data20 <- read.csv('dataset20.csv')
```

Call the `growth` function to estimate the model:

```
sem20 <- growth(model20, data=data20)
```

The `growth` function adds a mean structure model to estimate slope and intercept means.

Print the summary:

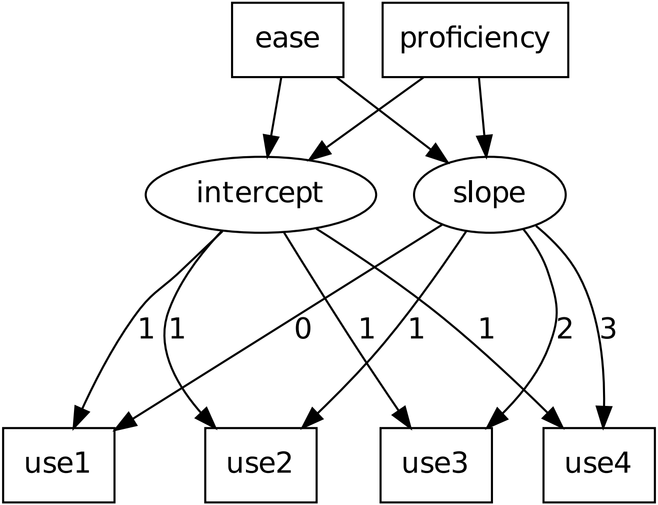
```
summary(sem20)
```

Model Specification

```
intercept =~ 1*use1 + 1*use2 + 1*use3 + 1*use4
slope =~ 0*use1 + 1*use2 + 2*use3 + 3*use4
```

```
Lavaan (0.4-9) converged normally after 78 iterations

  Number of observations                          2500

  Estimator                                         ML
  Minimum Function Chi-square                    6.171
  Degrees of freedom                                 5
  P-value                                        0.290

                   Estimate  Std.err  Z-value  P(>|z|)
Covariances:
  intercept ~~
    slope             0.497    0.024   21.020    0.000

Intercepts:
    intercept         1.219    0.021   58.259    0.000
    slope             1.446    0.021   70.440    0.000

Variances:
    intercept         1.090    0.031   35.227    0.000
    slope             1.052    0.030   35.318    0.000
```

Including External Covariates



Estimation in R

Read the model file:

```
model21 <- readLines('model21.txt')
```

Read the dataset:

```
data21 <- read.csv('dataset21.csv')
```

Call the `growth` function to estimate the model:

```
sem21 <- growth(model21, data=data21)
```

The `growth` function adds a mean structure model to estimate slope and intercept means.

Print the summary:

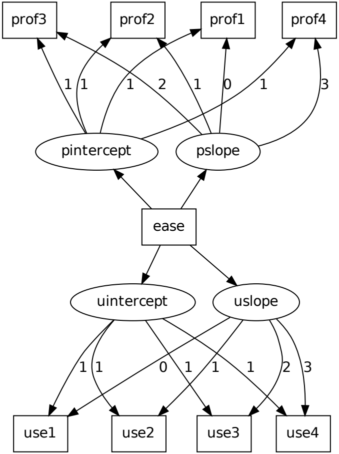
```
summary(sem21)
```

Model Specification

```
intercept =~ 1*use1 + 1*use2 + 1*use3 + 1*use4
slope =~ 0*use1 + 1*use2 + 2*use3 + 3*use4
intercept ~ ease + proficiency
slope ~ ease + proficiency
```

```
Lavaan (0.4-9) converged normally after 145 iterations

  Number of observations                          2500

  Estimator                                         ML
  Minimum Function Chi-square                   13.587
  Degrees of freedom                                 9
  P-value                                        0.138

                   Estimate  Std.err  Z-value  P(>|z|)
Regressions:
  intercept ~
    ease              0.995    0.002  411.252    0.000
    proficiency       0.248    0.002  103.435    0.000
  slope ~
    ease              0.252    0.003   79.050    0.000
    proficiency       0.997    0.003  315.664    0.000

Intercepts:
    intercept         0.005    0.004    1.193    0.233
    slope             0.256    0.005   47.593    0.000

Variances:
    intercept         0.010    0.000   24.411    0.000
    slope             0.023    0.001   33.686    0.000
```

Related Growth Curves



Estimation in R

Read the model file:

```
model22 <- readLines(’model22.txt’)
```

Read the dataset:

```
data22 <- read.csv(’dataset22.csv’)
```

Call the `growth` function to estimate the model:

```
sem22 <- growth(model22, data=data22)
```

Alternatively, read a manually specified model file:

```
model22a <- readLines(’model22a.txt’)
```

Use the `sem` function to estimate the model:

```
sem22a <- sem(model22a, data=data22)
```

Print the summaries:

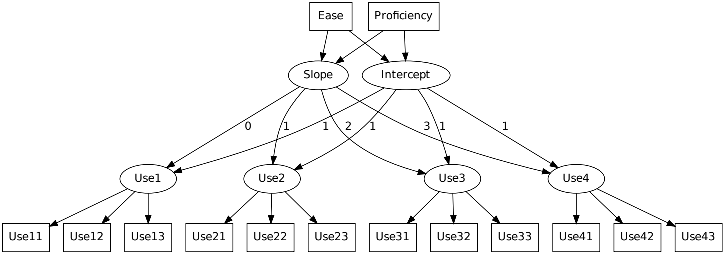
```
summary(sem22)
summary(sem22a)
```

Model Specification

```
# Growth curve 1
uintercept =~ 1*use1 + 1*use2 + 1*use3 + 1*use4
uslope =~ 0*use1 + 1*use2 + 2*use3 + 3*use4
uintercept ~ ease
uslope ~ ease
# Growth curve 2
pintercept =~ 1*prof1 + 1*prof2 + 1*prof3 + 1*prof4
pslope =~ 0*prof1 + 1*prof2 + 2*prof3 + 3*prof4
pintercept ~ ease
pslope ~ ease
# Freeing the intercepts for slopes/intercepts
uslope ~ NA*1
uintercept ~ NA*1
pintercept ~ NA*1
pslope ~ NA*1
# Fixing the intercepts for manifest variables
use1 ~ 0*1
use2 ~ 0*1
use3 ~ 0*1
use4 ~ 0*1
prof1 ~ 0*1
prof2 ~ 0*1
prof3 ~ 0*1
prof4 ~ 0*1
```

```
Lavaan (0.4-9) converged normally after 190 iterations

  Number of observations                          2500

  Estimator                                         ML
  Minimum Function Chi-square                   24.207
  Degrees of freedom                                26
  P-value                                        0.564

                   Estimate  Std.err  Z-value  P(>|z|)
Regressions:
  uintercept ~
    ease              0.251    0.002  104.951    0.000
  uslope ~
    ease              1.001    0.003  319.688    0.000
  pintercept ~
    ease             -0.003    0.004   -0.625    0.532
  pslope ~
    ease             -0.004    0.005   -0.863    0.388

Intercepts:
    uintercept       -0.002    0.003   -0.658    0.511
    uslope            0.251    0.004   57.125    0.000
    pintercept        0.006    0.006    1.071    0.284
    pslope            0.506    0.007   72.966    0.000
```

Latent Growth Curves



Estimation in R

Read the model file:

```
model23 <- readLines(’model23.txt’)
```

Read the dataset:

```
data23 <- read.csv(’dataset23.csv’)
```

Call the `growth` function to estimate the model:

```
sem23 <- growth(model23, data=data23)
```

Print the summary:

```
summary(sem23)
```

Model Specification

```
# standard stuff up front
use1 =~ use11 + a*use12 + b*use13
use2 =~ use21 + a*use22 + b*use23
use3 =~ use31 + a*use32 + b*use33
use4 =~ use41 + a*use42 + b*use43
# define intercept and slope as latent variables
# with the use latents as "indicators"
intercept =~ 1*use1 + 1*use2 + 1*use3 + 1*use4
slope =~ 0*use1 + 1*use2 + 2*use3 + 3*use4
# regress on ease and prof
intercept ~ ease + prof
slope ~ ease + prof
# factorial/measurement invariance constraints
use12 ~ c*1
use22 ~ c*1
use32 ~ c*1
use42 ~ c*1
use13 ~ d*1
use23 ~ d*1
use33 ~ d*1
use43 ~ d*1
```

```
Lavaan (0.4-9) converged normally after 290 iterations

  Number of observations                          2500

  Estimator                                         ML
  Minimum Function Chi-square                  103.059
  Degrees of freedom                                81
  P-value                                        0.050

                   Estimate  Std.err  Z-value  P(>|z|)
Regressions:
  intercept ~
    ease              1.003
    prof              0.501
  slope ~
    ease              0.503
    prof              0.995

Intercepts:
    intercept        -0.002
    slope             0.005

Variances:
    intercept         0.010
    slope             0.010
```

Bayesian SEM

- Bayesian SEM can incorporate existing knowledge about parameters using prior distributions
- Parameter vector θ is considered *random*:
  - *Prior distribution* $p(\theta|M)$ for a model M.
  - *Posterior distribution* $p(\theta|Y, M)$ for data Y and model M
- Joint distribution of data Y and parameter vector θ

$$p(Y, \theta|M) = p(Y|\theta, M)p(\theta|M) = p(\theta|Y, M)p(Y|M)$$

$$\Rightarrow \log p(\theta|Y, M) \propto \log p(Y|\theta, M) + \log p(\theta|M)$$

- Note that $p(Y|\theta, M)$ is the *likelihood function*
- Since $p(\theta)$ is not sample size dependent, it is dominated by the log-likelihood for large sample sizes, i.e. the prior probability matters less for larger samples.

Prior Distributions

- Non-informative priors contain no information, e.g. a constant, or large variance, etc.
- Informative priors contain information e.g. from prior studies, expert assessment, etc.
- Prior distributions have their own distribution parameters, called *hyperparameters*
- *Conjugate priors* imply that the posterior distribution will have the same parametric form

Conjugate Prior Example

- Assume a univariate binomial distribution

$$p(y|\theta) = \binom{n}{y} \theta^{y}(1-\theta)^{n-y}$$

- Consider an inverse beta distribution with hyperparameters α and β as prior

$$p(\theta) \propto \theta^{\alpha-1}(1-\theta)^{\beta-1}$$

- Then the posterior is of the form

$$\begin{aligned} p(\theta|Y) &\propto p(y|\theta)P(\theta) \\ &\propto \theta^{y}(1-\theta)^{n-y}\theta^{\alpha-1}(1-\theta)^{\beta-1} \\ &= \theta^{y+\alpha-1}(1-\theta)^{n-y-\beta-1} \end{aligned}$$

- This is an inverse beta distribution with hyperparameters $\alpha + y$ and $\beta + n - y$.

The Bayesian CFA Model

- Confirmatory factor analysis model

$$y_i = \Lambda \omega_i + \epsilon_i$$

- Assume that

$$\begin{aligned} \omega_i &\overset{D}{=} N[0, \Phi] \\ \epsilon_i &\overset{D}{=} N[0, \Psi_\epsilon] \end{aligned}$$

- Conjugate prior of $(\Lambda_k, \psi_{\epsilon k})$ is

$$\begin{aligned} \psi_{\epsilon k} &\overset{D}{=} \text{Inverted Gamma}(\alpha^*_{0\epsilon k}, \beta^*_{0\epsilon k}) \\ [\Lambda_k | \psi_{\epsilon k}] &\overset{D}{=} N\left[\Lambda_{0k}, \psi_{\epsilon k} H_{0yk}\right] \\ \Phi &\overset{D}{=} \text{Inverted Wishart}_q(R^*_0, \rho_0) \end{aligned}$$

Choosing Hyperparameters

- H_{0yk} expresses the (co-)variances of the prior distribution of Λ_k
  - Use small variances when prior is certain (e.g. $H = 0.5I$)
  - Use larger variances when prior knowledge is uncertain
- The inverted gamma distribution of $\psi_{\epsilon k}$ (the variance of ϵ_k) has mean $\frac{\beta}{\alpha-1}$ and variance $\frac{\beta^2}{(\alpha-1)^2(\alpha-2)^2}$.
  - Use a small mean/small variance when $\phi_{\epsilon k}$ (the variance of ϵ_k) is small and we are fairly certain (e.g. $\alpha_{0k} = 9, \beta_{0k} = 4$)
  - Use a large mean/large variance when $\phi_{\epsilon k}$ (the variance of ϵ_k) is large and we are not certain (e.g. $\alpha_{0k} = 6, \beta_{0k} = 10$)
- The mean of the inverted Wishart distribution of Φ is $\frac{R_0^{-1}}{\rho_0 - q - 1}$.
  - If we are fairly certain that Φ is not too far away from Φ_0, we can choose such that $R_0^{-1} = (\rho_0 - q - 1)\Phi_0$

Posterior Analysis

- Posterior distribution $p(\theta, Y) = p(Y|\theta)p(\theta)$ is typically analytically intractable (closed form)
  - Instead, simulate a large number of observations from $p(\theta|Y)$
- Difficulty handling latent variables
  - Treat random variables as missing data and augment observed data so that posteror distribution of the complete data set is easy to analyze.
- Markov Chain Monte Carlo using Gibbs sampling

Gibbs Sampling

- Decompose θ and Ω (latent variables) into components, then sample repeatedly:

$$
\begin{array}{rcl}
\theta_1^{(j+1)} & \text{from} & p(\theta_1|\theta_2^{(j)},\ldots,\theta_a^{(j)},\Omega^{(j)},Y) \\
\theta_2^{(j+1)} & \text{from} & p(\theta_2|\theta_1^{(j+1)},\ldots,\theta_a^{(j)},\Omega^{(j)},Y) \\
\vdots & \text{from} & \vdots \\
\theta_a^{(j+1)} & \text{from} & p(\theta_a|\theta_a^{(j+1)},\ldots,\theta_{a-1}^{(j+1)},\Omega^{(j)},Y) \\
\Omega_1^{(j+1)} & \text{from} & p(\Omega_1|\theta^{(j+1)},\Omega_2^{(j)},\ldots,\Omega_b^{(j)},Y) \\
\Omega_2^{(j+1)} & \text{from} & p(\Omega_2|\theta^{(j+1)},\Omega_1^{(j+1)},\ldots,\Omega_b^{(j)},Y) \\
\vdots & \text{from} & \vdots \\
\Omega_b^{(j+1)} & \text{from} & p(\Omega_b|\theta^{(j+1)},\Omega_1^{(j+1)},\ldots,\Omega_{b-1}^{(j+1)},Y)
\end{array}
$$

Gibbs Sampling

- $a + b$ steps in the jth iteration of the sampler
- At each step, each component in θ and Ω is updated conditional on the latest values of the other components
- Most of the distributions are normal, gamma, or Wishart distribution, easily simulated
- Convergence at exponential rate after J ”burn-in” iterations

CFA Example

Read the model file:

```
model24 <- readLines(’model24.txt’)
```

Read the data:

```
data24 <- read.csv(’dataset24.txt’)
```

Call the `sem` function to estimate the model:

```
sem24 <- sem(model24, data=data24)
```

Print the summary, including standardized estimates:

```
summary(sem24)
```

```
  eou =~
    eou1              1.000
    eou2              1.044    0.007  152.931    0.000
    eou3              0.941    0.008  111.362    0.000
  use =~
    use1              1.000
    use2              0.947    0.006  148.832    0.000
    use3              0.891    0.007  130.496    0.000
  int =~
    int1              1.000
    int2              1.000    0.001 1338.017    0.000
    int3              1.000    0.001 1836.214    0.000
Covariances:
  eou ~~
    use               0.750    0.048   15.678    0.000
    int               1.579    0.100   15.739    0.000
  use ~~
    int               1.677    0.106   15.771    0.000
Variances:
    eou1              0.010    0.001   13.320    0.000
    eou2              0.006    0.001   10.862    0.000
    eou3              0.017    0.001   14.591    0.000
    use1              0.005    0.001   10.670    0.000
    use2              0.011    0.001   13.994    0.000
    use3              0.014    0.001   14.581    0.000
    int1              0.000    0.000    3.156    0.002
    int2              0.001    0.000   13.853    0.000
    int3              0.000    0.000   10.492    0.000
    eou               0.711    0.046   15.599    0.000
    use               0.801    0.051   15.707    0.000
    int               3.520    0.223   15.811    0.000
```

CFA Example using `MCMCpack`

- `MCMCpack` allows hyperparameters for
  - The prior of $\Lambda \overset{D}{=} N(\Lambda_0, \psi_{\epsilon k} H_0)$
  - The prior of the error variances $\Psi_\epsilon \overset{D}{=} IG(\alpha_0, \beta_0)$.
- Assume the following prior knowledge:
  - Means of the loadings are 0.8 with high certainty $\Rightarrow$ `l0=0.8, L0=10`
  - Error variances are large, but uncertain (high mean, high variance) $\Rightarrow$ `a0=6, b0=10`

- `MCMCfactanal` allows specification of loadings and cross-loadings. In our CFA model, we wish to restrict cross-loadings to 0, and scale by fixing the first loading to 1

```
constraints <- list(
  eou1=list(1, 1),
  eou1=list(2, 0),
  eou1=list(3, 0),
  eou2=list(2, 0),
  eou2=list(3, 0),
  eou3=list(2, 0),
  eou3=list(3, 0),
  use1=list(2, 1),
  use1=list(1, 0),
  use1=list(3, 0),
  use2=list(1, 0),
  use2=list(3, 0),
  use3=list(1, 0),
  use3=list(3, 0),
  int1=list(3, 1),
  int1=list(1, 0),
  int1=list(2, 0),
  int2=list(1, 0),
  int2=list(2, 0),
  int3=list(1, 0),
  int3=list(2, 0))
```

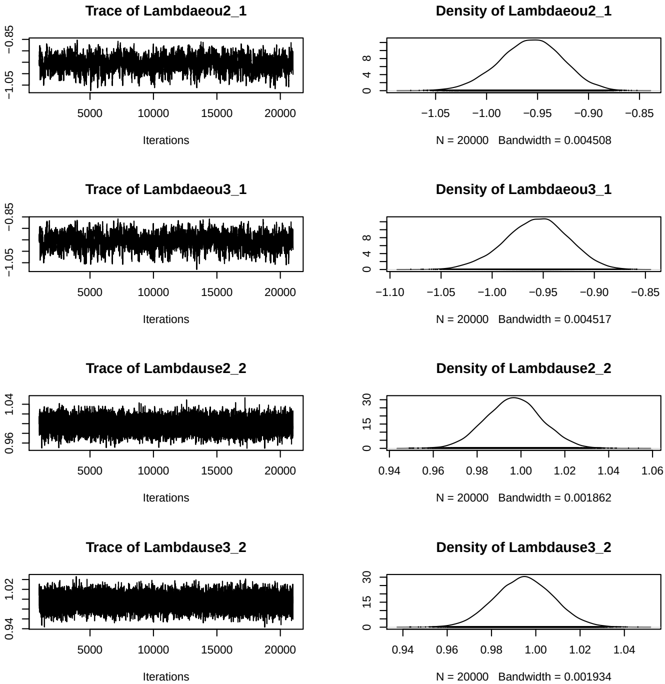
CFA Example using `MCMCpack`

```
library(MCMCpack)

f <- MCMCfactanal(as.matrix(data24),
                  factors=3,
                  lambda.constraints=constraints,
                  l0=0.5, L0=10,
                  a0=6, b0=10)

summary(f)

plot(f)
```

- Parameter `l0` may be a vector to specify different prior means
- Parameter `L0` is the inverse variance (”precision”) and may be a matrix to specify different prior variances

```
Iterations = 1001:21000
Thinning interval = 1
Number of chains = 1
Sample size per chain = 20000

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

                Mean       SD  Naive SE Time-series SE
Lambdaeou2_1 -0.95680 0.030827 2.180e-04      1.270e-03
Lambdaeou3_1 -0.95665 0.030888 2.184e-04      1.264e-03
Lambdause2_2  0.99681 0.012771 9.030e-05      1.876e-04
Lambdause3_2  0.99479 0.013223 9.350e-05      1.762e-04
Lambdaint2_3  0.99900 0.010816 7.648e-05      1.526e-04
Lambdaint3_3  0.99906 0.010819 7.650e-05      1.628e-04
Psieou2       0.05626 0.004657 3.293e-05      4.787e-05
Psieou3       0.05622 0.004664 3.298e-05      5.794e-05
Psiuse1       0.04090 0.002997 2.119e-05      2.445e-05
Psiuse2       0.04397 0.003238 2.290e-05      3.247e-05
Psiuse3       0.04807 0.003553 2.512e-05      3.186e-05
Psiint1       0.02976 0.002048 1.448e-05      1.724e-05
Psiint2       0.02982 0.002024 1.431e-05      1.425e-05
Psiint3       0.02979 0.002045 1.446e-05      1.461e-05
```

```
2. Quantiles for each variable:

                 2.5%      25%      50%      75%    97.5%
Lambdaeou2_1 -1.01951 -0.97739 -0.95606 -0.93544 -0.89806
Lambdaeou3_1 -1.02009 -0.97706 -0.95584 -0.93545 -0.89827
Lambdause2_2  0.97190  0.98817  0.99677  1.00523  1.02201
Lambdause3_2  0.96901  0.98586  0.99477  1.00375  1.02087
Lambdaint2_3  0.97803  0.99177  0.99894  1.00614  1.02056
Lambdaint3_3  0.97807  0.99163  0.99903  1.00637  1.02017
Psieou2       0.04778  0.05304  0.05605  0.05924  0.06608
Psieou3       0.04766  0.05297  0.05600  0.05920  0.06588
Psiuse1       0.03546  0.03882  0.04075  0.04285  0.04716
Psiuse2       0.03806  0.04172  0.04382  0.04607  0.05076
Psiuse3       0.04161  0.04561  0.04792  0.05035  0.05546
Psiint1       0.02598  0.02832  0.02967  0.03109  0.03394
Psiint2       0.02613  0.02841  0.02974  0.03114  0.03404
Psiint3       0.02608  0.02836  0.02970  0.03112  0.03405
```


Trace of Lambdaeou2_1
−0.85
−1.05
5000
10000
15000
20000
Iterations
Density of Lambdaeou2_1
0
4
8
−1.05
−1.00
−0.95
−0.90
−0.85
N = 20000 Bandwidth = 0.004508
Trace of Lambdaeou3_1
−0.85
−1.05
5000
10000
15000
20000
Iterations
Density of Lambdaeou3_1
0
4
8
−1.10
−1.05
−1.00
−0.95
−0.90
−0.85
N = 20000 Bandwidth = 0.004517
Trace of Lambdause2_2
1.04
0.96
5000
10000
15000
20000
Iterations
Density of Lambdause2_2
0
15
30
0.94
0.96
0.98
1.00
1.02
1.04
1.06
N = 20000 Bandwidth = 0.001862
Trace of Lambdause3_2
1.02
0.94
5000
10000
15000
20000
Iterations
Density of Lambdause3_2
0
15
30
0.94
0.96
0.98
1.00
1.02
1.04
N = 20000 Bandwidth = 0.001934

MCMC Diagnostics

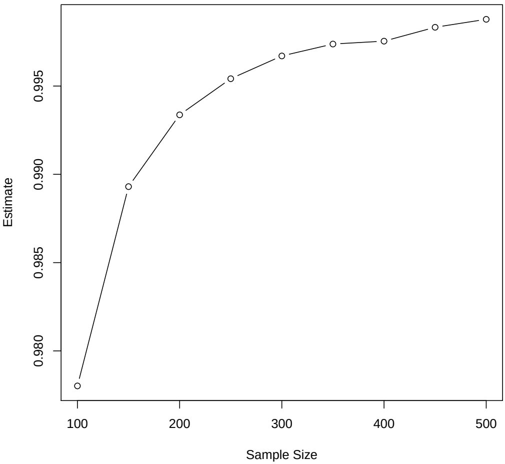
- Check convergence (test of equality of means at the beginning and end of MCMC chain)

```
library(coda)
geweke.diag(f)

Fraction in 1st window = 0.1
Fraction in 2nd window = 0.5

Lambdaeou2_1 Lambdaeou3_1 Lambdause2_2 Lambdause3_2 Lambdaint2_
    -0.37911     -0.25174      0.66345     -0.21312     -0.252
     Psieou2      Psieou3      Psiuse1      Psiuse2      Psius
    -0.90353     -0.56351     -0.19265      1.96520      0.298
     Psiint2      Psiint3
    -0.62875      0.35001
```

- Check convergence (Cramer-von-Mises test of stationary distribution)

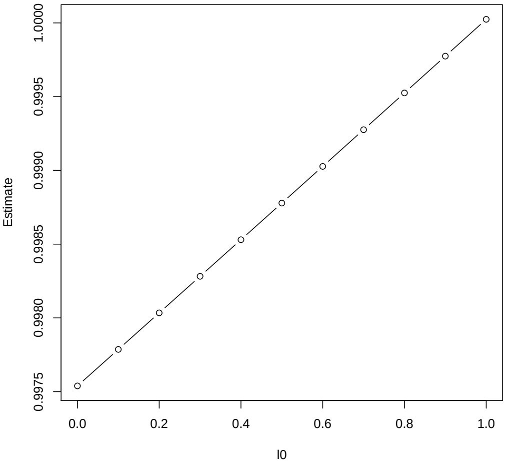
```
heidel.diag(f)
             Stationarity start     p-value
             test         iteration
Lambdaeou2_1 passed       1         0.539
Lambdaeou3_1 passed       1         0.509
Lambdause2_2 passed       1         0.301
Lambdause3_2 passed       1         0.320
Lambdaint2_3 passed       1         0.361
Lambdaint3_3 passed       1         0.295
Psieou2      passed       1         0.639
Psieou3      passed       1         0.347
Psiuse1      passed       1         0.965
Psiuse2      passed       1         0.127
Psiuse3      passed       1         0.238
Psiint1      passed       1         0.491
Psiint2      passed       1         0.908
Psiint3      passed       1         0.808
```

Sample Size



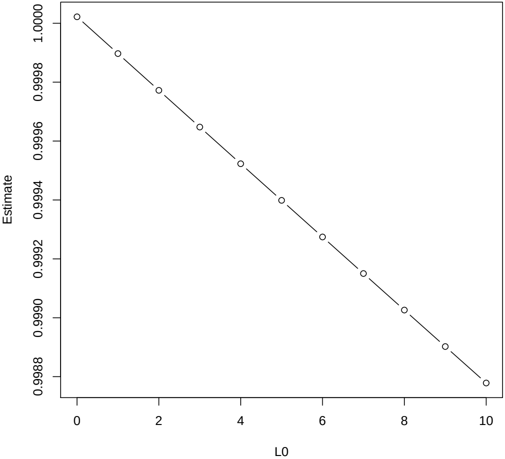
(l0=0.5, L0=10, a0=6, b0=10)

Prior Means



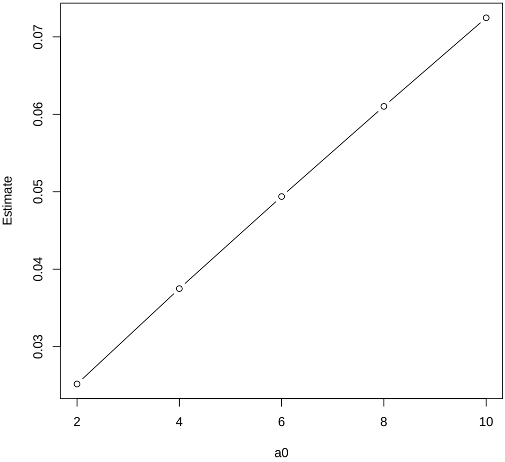
(sample size=500, L0=10, a0=6, b0=10)

Prior Precision (Inverse Variance)



(sample size=500, l0=0.8, a0=6, b0=10)

Prior Shape Parameter (for IG of errors)



(sample size=500, l0=0.8, L0=10, b0=2*a0)

Latent Variable	*Indicators*	
Continuous	Continuous	Factor Model
Categorical	Categorical	Latent Class Analysis
Continuous	Categorical	Item Response Model
Categorical	Continuous	Latent Profile Analysis

LCA Model

- Number of classes C
- Latent class prevalences γ_c (posterior probability of membership in class)
- Item-response probabilities ρ_j (probability of item responses conditional on class membership)

$$P(Y = y) = \sum_{c=1}^{C} \gamma_c \prod_{j=1}^{J} \prod_{r_j=1}^{R_J} \rho_{j,r_j|c}^{I(y_j=r_j)}$$

- Parameter estimation using ML via EM algorithm

Number of items J, Number of categories/answers for each question R_j

Evaluating LCA Results

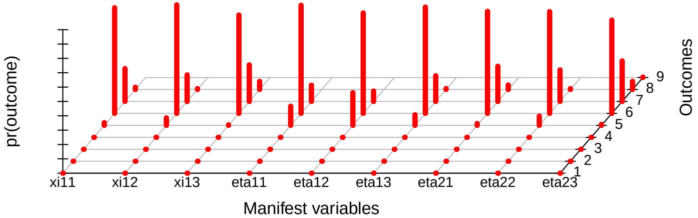
- G^2 likelihood ratio statistic (smaller is better)
  - Difference in G^2 statistics of nested models are $\approx \chi^2$ distributed for significance testing
- AIC and BIC information criteria (smaller is better)
- Homogeneity of response patterns (probabilities) within classes
- Separation of classes through distinct response patterns (probabilities)

LCA in R using the poLCA package

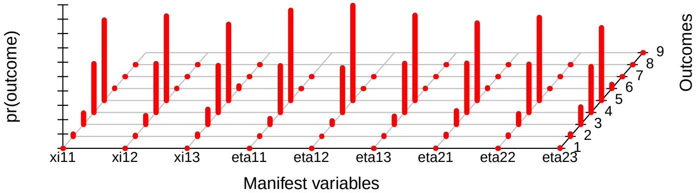
```
library(poLCA)

data <- read.csv(’dataset25’)

lca.result <- poLCA(cbind(xi11, xi12, xi13,
                         eta11, eta12, eta13,
                         eta21, eta22, eta23) ~1,
                    data=data+abs(min(data))+1,
                    nclass=2)

lca.result
plot(lca.result)
```

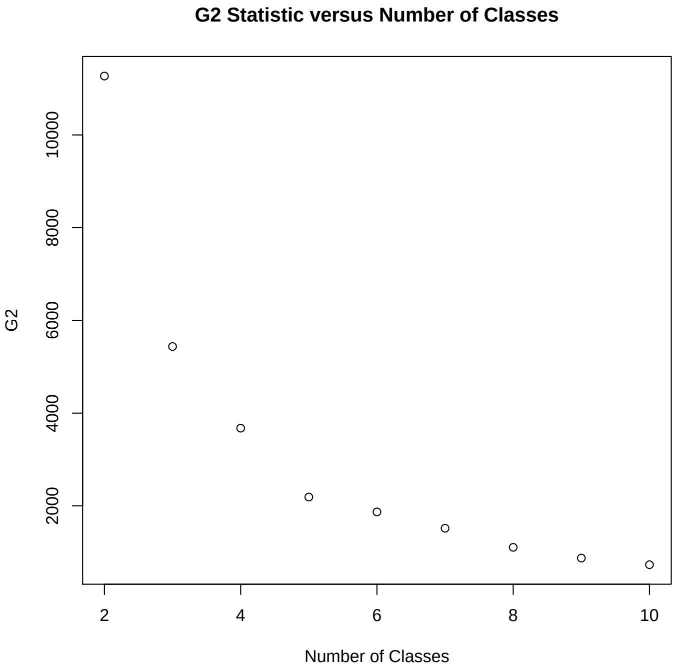
```
Conditional item response (column) probabilities,
 by outcome variable, for each class (row)

$xi11
          Pr(1)  Pr(2)  Pr(3)  Pr(4)  Pr(5)  Pr(6)  Pr(7)  Pr(8)
class 1:      0 0.0000 0.0000 0.0000 0.0188 0.7363 0.2292 0.0157
class 2:      0 0.0161 0.0792 0.3404 0.5606 0.0036 0.0000 0.0000

...

$eta23
           Pr(1)  Pr(2)  Pr(3)  Pr(4)  Pr(5)  Pr(6)  Pr(7)  Pr(8)  Pr(9)
class 1:  0.0000 0.0000 0.0000 0.0000 0.0075 0.6503 0.2825 0.0565 0.0031
class 2:  0.0044 0.0176 0.1218 0.3199 0.5072 0.0291 0.0000 0.0000 0.0000

Estimated class population shares
 0.3185 0.6815

Predicted class memberships (by modal posterior prob.)
 0.318 0.682

=========================================================
Fit for 2 latent classes:
=========================================================
number of observations: 1000
number of estimated parameters: 125
residual degrees of freedom: 875
maximum log-likelihood: -8601.77

AIC(2): 17453.54
BIC(2): 18067.01
G^2(2): 11221.36 (Likelihood ratio/deviance statistic)
X^2(2): 3.362642e+18 (Chi-square goodness of fit)
```

Class 1: population share = 0.319



Class 2: population share = 0.681




G2 Statistic versus Number of Classes
G2
2000
4000
6000
8000
10000
2
4
6
8
10
Number of Classes

Class Assignment

- Inspect Class Assignment

```
lca.result$predclass
```

- Compare class assignment differences to actual group

```
sum(abs(lca.result$predclass - data$Group))
```

Use for Multi-Group SEM

- Groups formed by LCA ($\chi^2_{48} = 1046.910$)

```
model <- readLines('model25.txt')
data <- cbind(data, lca.result$predclass)
colnames(data)[11] <- 'LCAGroup'
s <- sem(model, data=data, group='LCAGroup')
```

- Compare to no groups ($\chi^2_{24} = 975.392$)

```
s <- sem(model, data=data)
```

- Compare to original groups ($\chi^2_{48} = 999.086$)

```
s <- sem(model, data=data, group='Group')
```

Resources

- Books
  - Bollen (1989)
  - Hayduk (1987, 1996)
  - Kaplan (2000)
  - Mulaik (2009)
  - Lee (2007)
  - Collins and Lanza (2010)
- Journals
  - Structural Equation Modeling
  - Psychological Methods
  - Organizational Research Methods
  - Multivariate Behavioral Research
  - ...
- Web
  - SEMNet mailing list (`http://aime.ua.edu/cgi-bin/wa?A0=SEMNET`)