CFA (SEM) and EFA

Lin Wenzheng

2021/12/2

SEM Introduction

- SEM represents a family of related procedures.
- It is also known as covariance structural analysis, covariance structure model, analysis of covariance structure, analysis of correlation structure, LISREL model (in the old days), and causal modeling (not preferred nowadays).

Observed vs. latent variables

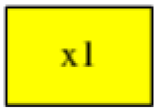
- Latent variables:
  - Abstract and hypothetical constructs.
  - They cannot be measured directly.
  - Most constructs in psychology and social sciences are unobservable.
    - For example, motivation, stress, depression, intelligence and satisfaction.
  - What is the common strategy to measure psychological constructs in psychology?
- Observed or measured variables:
  - Indicators of the latent constructs.
    - For example, items to measure depression, test scores of intelligence.
  - How many items do we need to measure depression?

Models of relationship among the constructs

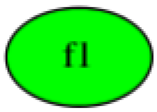
- Most statistical techniques are limited to one dependent variable (DV).
- SEM allows researchers to test models with a complicated relationship.
  - Models are usually represented by path diagrams.
- Model fitting:
  - Overall model fit: Does the proposed model fit the data?
  - Individual fit: Is the parameter estimate statistically significant?

Path diagram

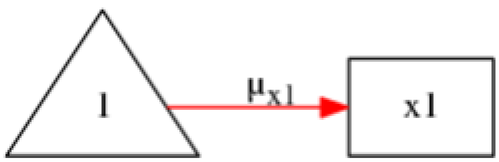
- Rectangles or squares: observed (or measured) variables



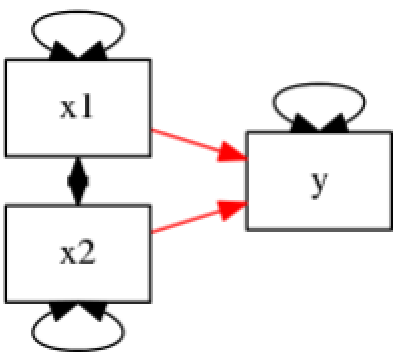
- Circles or ellipses: unobserved (or latent) variables



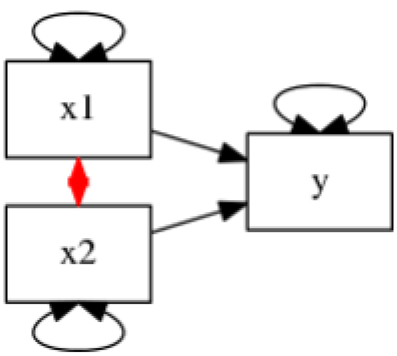
- Triangles: means or intercepts



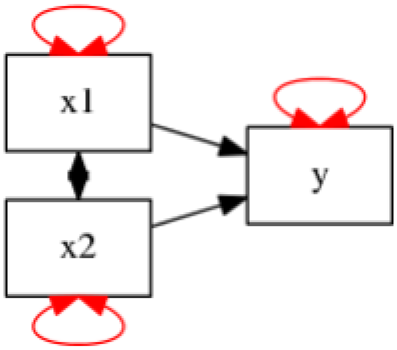
- Direct arrows: predictions or "causes"



- Double arrows between two variables: covariances



- Double arrows on the same variables: variances



SEM: Covariance matrix as inputs

- Variance and covariance matrix
  - The covariance matrix plays an important role is SEM.
  - When the data are multivariate normal, the means and covariance matrix are the sufficient statistics. That is, we do not need the raw data in the analysis.
- Some useful formulas
  - $var(x) = var(x_1 + x_2 + ... + x_k) = \sum_{i=1}^{k} var(x_i) + 2\sum_{i=1}^{k}\sum_{j=1}^{i} cov(x_i, x_j)$
  - $cov(x, y) = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{n}$
  - $cov(x_i + x_j, x_s + x_t) = cov(x_i, x_s) + cov(x_i, x_t) + cov(x_j, x_s) + cov(x_j, x_t)$
  - $cor(x_i, x_j) = \frac{cov(x_i, x_j)}{\sqrt{var(x_i)var(x_j)}}$
  - $var(k * x) = k^2 + var(x)$ where k is a constant
  - $var(k + x) = var(x)$

A composite score

- Suppose x is a composite score, x = x1 + x2 + x3.
- $$S = \begin{array}{c} \\ x1 \\ x2 \\ x3 \end{array} \begin{array}{c} \begin{array}{ccc} x1 & x2 & x3 \end{array} \\ \begin{bmatrix} 1.6111111 & 0.6666667 & 0.2222222 \\ 0.6666667 & 1.8222222 & 0.4444444 \\ 0.2222222 & 0.4444444 & 0.4444444 \end{bmatrix} \end{array}$$
- $$R = \begin{array}{c} \\ x1 \\ x2 \\ x3 \end{array} \begin{array}{c} \begin{array}{ccc} x1 & x2 & x3 \end{array} \\ \begin{bmatrix} 1.0000000 & 0.3890857 & 0.2626129 \\ 0.3890857 & 1.0000000 & 0.4938648 \\ 0.2626129 & 0.4938648 & 1.0000000 \end{bmatrix} \end{array}$$
  - What is var(x)?

- What is $\text{cov}(x_1 + x_2, x_3)$?
- What is $\text{cor}(x_1, x_2)$?

Covariance matrix

- Covariance matrix is usually used as the input in SEM.
- The covariance matrix of y, x1 and x2 is a 3x3 symmetric matrix:

An SEM example

```
my.df <- read.csv("SEM01.csv")
head(my.df)
```

```
##   X    y   x1   x2
## 1 1 2.74 6.86 1.57
## 2 2 3.19 4.72 0.26
## 3 3 3.21 8.55 0.51
## 4 4 5.22 8.88 3.45
## 5 5 2.03 9.81 2.43
## 6 6 5.45 8.55 1.11
```

```
tail(my.df)
```

```
##       X    y    x1   x2
## 95   95 5.05  7.28 1.92
## 96   96 5.89 10.08 2.72
## 97   97 3.73  6.65 2.15
## 98   98 5.62 10.14 1.60
## 99   99 3.34  7.67 0.54
## 100 100 2.38  6.83 0.11
```

```
cov(my.df)
```

```
##              X         y         x1         x2
## X  841.6666667 0.1052525 -1.1343939 -0.2171212
## y    0.1052525 2.1653210  1.8855177  0.7902861
## x1  -1.1343939 1.8855177  3.6882648  0.9638535
## x2  -0.2171212 0.7902861  0.9638535  1.4297986
```

```
cor(my.df)
```

```
##               X           y          x1           x2
## X   1.000000000 0.002465479 -0.02036024 -0.006258853
## y   0.002465479 1.000000000  0.66720369  0.449144199
## x1 -0.020360235 0.667203686  1.00000000  0.419722778
## x2 -0.006258853 0.449144199  0.41972278  1.000000000
```

```
library(lavaan)
```

```
## Warning: package 'lavaan' was built under R version 4.1.2
```

```
## This is lavaan 0.6-9
## lavaan is FREE software! Please report any bugs.
```

```
## Build a model
## "~~" represents variance or covariance
my.model <- 'y ~~ x1 # covariance
y ~~ y # variance of y (optional)
x1 ~~ x1 # variance of x1 (optional)'

## Fit the model
## We do not need x2 in this example
my.fit <- sem(my.model, data=my.df)

## Show the results
summary(my.fit)
```

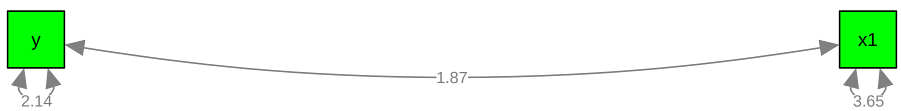
```
## lavaan 0.6-9 ended normally after 13 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                         3
##
##   Number of observations                           100
##
## Model Test User Model:
##
##   Test statistic                                 0.000
##   Degrees of freedom                                 0
##
## Parameter Estimates:
##
##   Standard errors                             Standard
##   Information                                 Expected
##   Information saturated (h1) model          Structured
##
## Covariances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   y ~~
##     x1                1.867    0.336    5.550    0.000
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##     y                 2.144    0.303    7.071    0.000
##     x1                3.651    0.516    7.071    0.000
```

```r
library(semPlot)
```

```
## Warning: package 'semPlot' was built under R version 4.1.2
```

```r
## Plot the model without parameter estimates
semPaths(my.fit, what="path")
```


```r
## Plot the parameter estimates
semPaths(my.fit, what="est", color="green", weighted=FALSE)
```



Interpretation

- SEM is usually based on maximum likelihood (ML) estimation method. It is asymptotically unbiased (when the sample size is getting larger and larger).
- The parameter estimates, especially the variances, can be biased when the sample sizes are small.
- Thus, the estimates are slightly different from those reported in descriptive statistics.

Comparison with simple regression

```
my.lm <- lm(y~x1,data = my.df)
summary(my.lm)
```

```
##
## Call:
## lm(formula = y ~ x1, data = my.df)
##
## Residuals:
```

```
##      Min       1Q   Median       3Q      Max
## -2.52988 -0.66877 -0.02702  0.78679  2.36542
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.45519    0.46309  -0.983    0.328
## x1           0.51122    0.05765   8.867 3.47e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.102 on 98 degrees of freedom
## Multiple R-squared:  0.4452, Adjusted R-squared:  0.4395
## F-statistic: 78.63 on 1 and 98 DF,  p-value: 3.467e-14
```

- A simple regression specifies the direction of influence.
- Population model: $y = b_0 + b_1 x_1 + e_y$
- Prediction model: $\hat{y} = \hat{b}_0 + \hat{b}_1 x_1$.
- Variance decomposition: $var(y) = b_{12} var(x_1) + var(e_y)$. Why?
- Since y is a dependent variable, var(y) depends on other parameters. This means that the double arrows in x1 and y have different meanings depending on whether they are independent or dependent variables.

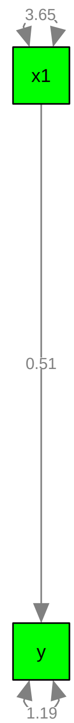
```
## Build a model
my.model <- 'y ~ 1 # intercept (optional)
y ~ x1 # regression coefficient
y ~~ y # error variance of y (optional)'

## Fit the model
my.fit <- sem(my.model, data=my.df)

## Show the results
summary(my.fit)
```

```
## lavaan 0.6-9 ended normally after 13 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                         3
##
##   Number of observations                           100
##
## Model Test User Model:
##
##   Test statistic                                 0.000
##   Degrees of freedom                                 0
##
## Parameter Estimates:
##
##   Standard errors                             Standard
##   Information                                 Expected
```

```
##   Information saturated (h1) model          Structured
##
## Regressions:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   y ~
##     x1                0.511    0.057    8.957    0.000
##
## Intercepts:
##                    Estimate  Std.Err  z-value  P(>|z|)
##    .y                -0.455    0.458   -0.993    0.321
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##    .y                 1.189    0.168    7.071    0.000
```

```r
## Plot the model without parameters
semPaths(my.fit, what="path", intercepts=FALSE)
```


```r
## Plot the parameter estimates
semPaths(my.fit, what="est", color="green", intercepts=FALSE, weighted=FALSE)
```



Interpretations

- The estimated regression equation is $\hat{y} = -0.455 + 0.511 * x_1$.
- The parameter estimates divided by their corresponding standard errors (SEs) approximately follow a standard norm distribution. If the absolute values are larger than 1.96, they are statistically significant at $= .05$. Unfortunately, this approach may not be accurate in some cases. We will address this issue in the later lectures.

A multiple regression

- A multiple regression may also be formulated as a structural equation model.
- Population model: $y = b_0 + b_1 x_1 + b_2 x_2 + e_y$
- Prediction model: $\hat{y} = \hat{b}_0 + \hat{b}_1 x_1 + \hat{b}_2 x_2$
- Variance decomposition: $var(y) = b_{12} var(x_1) + b_{22} var(x_2) + 2 b_1 b_2 cov(x_1, x_2) + var(e_y)$.

```r
## Regression
summary( lm(y~x1+x2, data=my.df) )
```

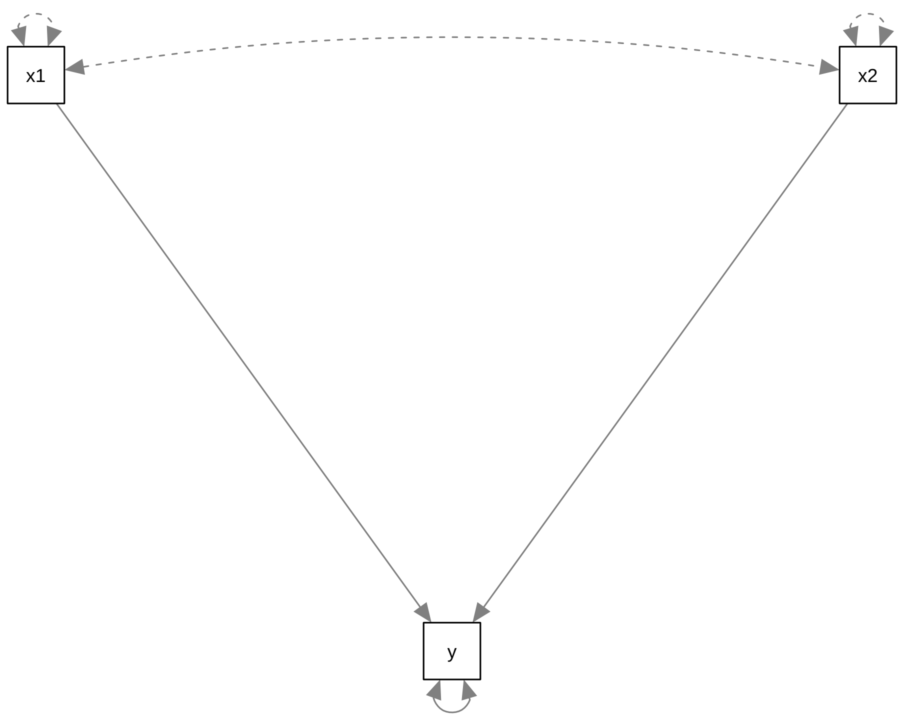
```
##
## Call:
## lm(formula = y ~ x1 + x2, data = my.df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.63549 -0.74128  0.03412  0.76019  2.24242
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.31583    0.45399  -0.696   0.4883
## x1           0.44521    0.06182   7.202 1.28e-10 ***
## x2           0.25260    0.09928   2.544   0.0125 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.072 on 97 degrees of freedom
## Multiple R-squared:  0.4799, Adjusted R-squared:  0.4691
## F-statistic: 44.75 on 2 and 97 DF,  p-value: 1.704e-14
```

```r
## Build a model
my.model <- 'y ~ 1 # intercept
y ~ x1+x2 # regression coefficients
y ~~ y # error variance of y'

## Fit the model
my.fit <- sem(my.model, data=my.df)

## Show the results
summary(my.fit)
```

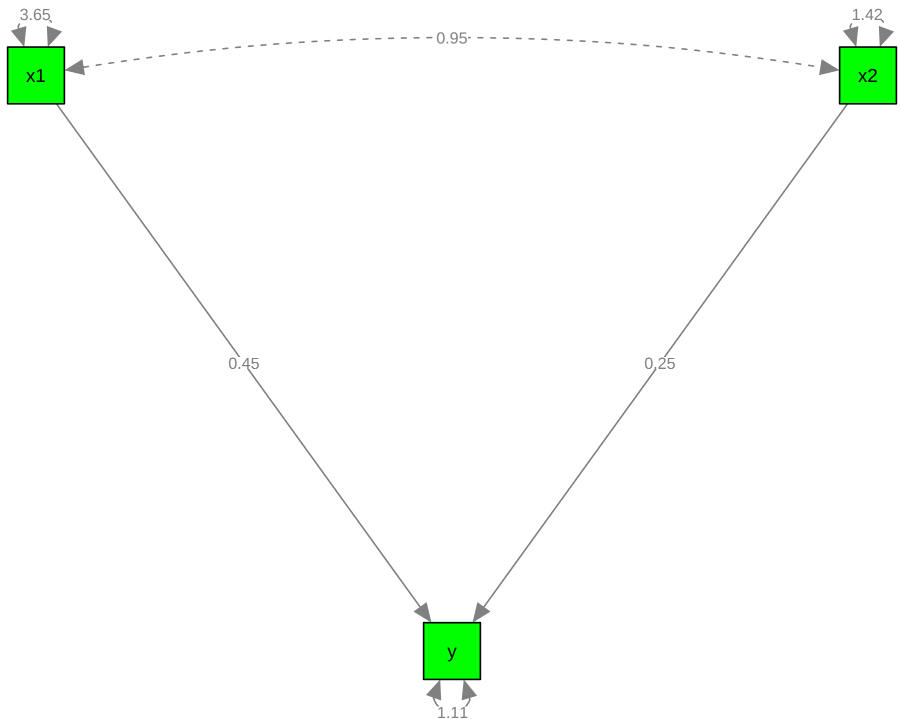
```
## lavaan 0.6-9 ended normally after 13 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                         4
##
##   Number of observations                           100
##
## Model Test User Model:
##
##   Test statistic                                 0.000
##   Degrees of freedom                                 0
##
## Parameter Estimates:
##
##   Standard errors                             Standard
##   Information                                 Expected
##   Information saturated (h1) model          Structured
##
## Regressions:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   y ~
##     x1                0.445    0.061    7.313    0.000
```

```
##     x2                0.253    0.098    2.583    0.010
##
## Intercepts:
##                    Estimate  Std.Err  z-value  P(>|z|)
##    .y                -0.316    0.447   -0.706    0.480
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##    .y                 1.115    0.158    7.071    0.000
```

```r
## Plot the model
semPaths(my.fit, what="path", intercepts=FALSE)
```




```r
## Plot the parameter estimates
semPaths(my.fit, what="est", color="green", intercepts=FALSE, weighted=FALSE)
```



Factor analysis

What is factor analysis for?

- Factor analysis can be utilized to examine the underlying patterns or relationships for a large number of variables and to determine whether the information can be condensed or summarized in a smaller set of factors or components (Hair et al., 2006, p. 101).

Examples in psychological research

- Intelligence
  - Three general factors (e.g., Kline, 2000);
  - General intelligence (g) introduced by Spearman, fluid and crystallized intelligence;
  - Wechsler Adult Intelligence Scale-Third Version (WAIS-III):
    - General intelligence;
    - Verbal IQ and performance IQ.

- Personality
  - Five-factor Model (e.g., Costa & McCrae, 1992):
    * OCEAN: Openness to experience/Conscientiousness/Extraversion/Agreeableness/Neuroticism
    * Conceptualized as independent.
  - Higher-order factors (alpha and beta) and general factor of personality
- Factor analysis is frequently used to validate the measurements in social sciences.

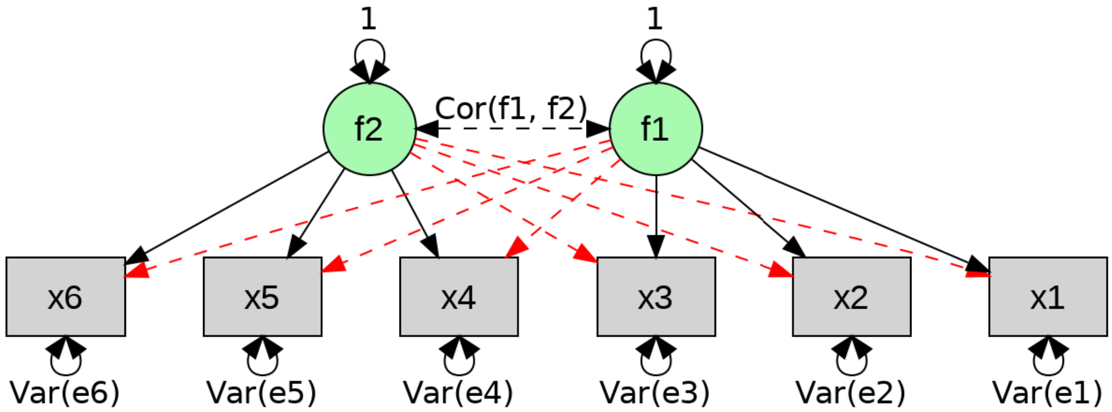
Exploratory factor analysis (EFA) vs. Confirmatory factor analysis (CFA)

EFA	CFA
Exploratory nature for theory development	Confirmatory nature for theory testing
No. of factors are usually determined from data	No. of factors are specified a priori
Factor loading pattern are not fixed	Factor loading pattern are fixed
Using factor rotation to achieve simple structure	Fixing factor variances/loadings for identification
Factors are usually orthogonal	Factors are usually correlated
Measurement errors are uncorrelated	Measurement errors can be correlated
A few statistical tests for model testing	Formal statistical tests and goodness-of-fit indices
Correlation matrix as input	Covariance matrix as input

Exploratory factor analysis (EFA)

Introduction

- Data
  - x: continuous variables- observed, measured or indicator variables;
  - f : continuous variables- unobserved, latent or common factors.
  - Binary variables are usually not appropriate for EFA. Special programs, e.g., Mplus, may be required.
  - Correlation is usually used in the analysis.
- Purposes of factor analysis
  - Theory development:
    * Test and explore the internal structure of the questionnaires (construct validity).
  - Data simplification:
    * Create a small no. of composite scores for further analysis.
    * Reduce the multicollinearity among the predictors in multiple regression.

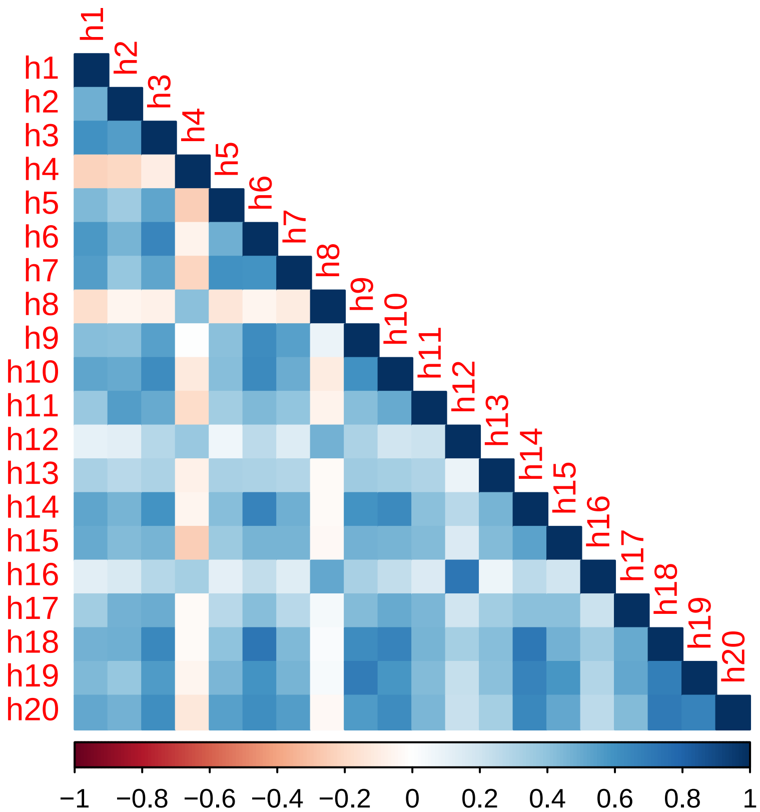


```
my.df3 <- read.csv("EFA03.csv")
ma.df3 <- cor(my.df3)
library(corrplot)
```

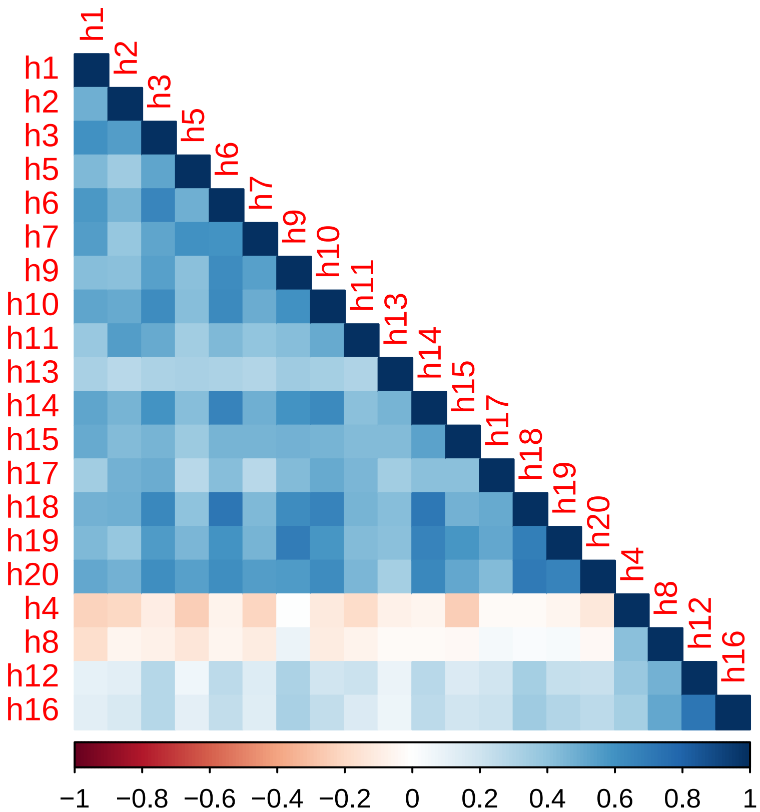
```
## Warning: package 'corrplot' was built under R version 4.1.2
```

```
## corrplot 0.92 loaded
```

```
corrplot(ma.df3, method="color", type="lower")
```

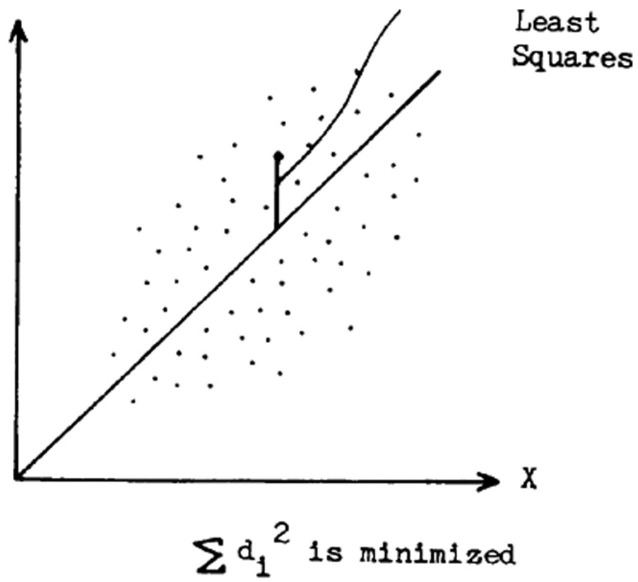


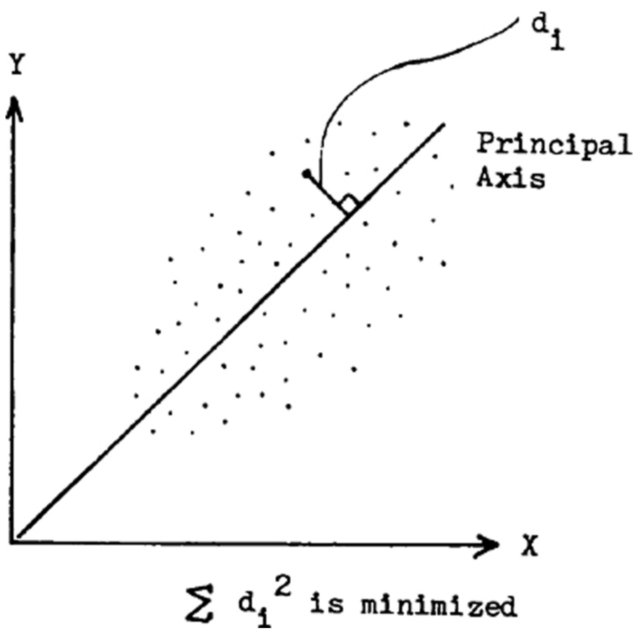

```r
my.df3.1 <- my.df3[, !colnames(my.df3) %in% c("h4","h8","h12","h16")]
my.df3.1 <- cbind(my.df3.1,my.df3[,colnames(my.df3) %in% c("h4","h8","h12","h16")])
ma.df3.1 <- cor(my.df3.1)
corrplot(ma.df3.1, method="color", type="lower")
```



An two-factor example

- $x = \Lambda * f + e$,
  - x is the $p \times 1$ observed variables,
  - Λ is the $p \times k$ factor loading matrix,
  - f is the $k \times 1$ common factor, and
  - e is the $p \times 1$ unique factor or measurement error.
- $x_1 = {}_{11}f_1 + {}_{12}f_2 + e_1$
- $x_2 = {}_{21}f_1 + {}_{22}f_2 + e_2$
- $\vdots$
- $x_6 = {}_{61}f_1 + {}_{62}f_2 + e_6$
- What is the maximum number of factors in this model?





Standard assumptions in EFA

- Common factors and errors are uncorrelated: $Cov(f, e) = 0$;
- Measurement errors are uncorrelated: Cov(e1, e2) = 0;
- Means of f and e are zero: E(f) = E(e) = 0.
- Following these assumptions, the model for the population correlation matrix is
  - If factors are correlated: $\Sigma = \Lambda\Phi\Lambda^{\mathrm{T}} + \Psi$.
  - If factors are uncorrelated: $\Sigma = \Lambda\Lambda^{\mathrm{T}} + \Psi$.

- Σ is a p $\times$p population correlation matrix.
- Φ is a k $\times$k factor correlation matrix.
- Λ is a p $\times$k factor loading matrix.
- Ψ is a p $\times$p error variance matrix.

Four major steps in EFA

- Data collection and preparation: Are the data appropriate for EFA?
- Extraction of initial factors: How many factors are required?
- Factor rotation and interpretations: Are the factors correlated? What are the meanings of the factors?
- Estimation of factor scores for further analysis: Do we need to create factor scores for further analysis?

Data requirements

- Input data: continuous variables.
- It is suggested that the subject/variable ratio should be at least 10.
- Testing the factorability
  - Variables should be correlated;
  - If the variables are uncorrelated, silly and misleading findings can be obtained.
- Bartlett's test of sphericity
  - To test whether all variables are independent, H0 : $_{ij} = 0$ for all ith and jth variables.
  - If it is not rejected, it is not appropriate for EFA.
  - The test is sensitive to the sample size.
- Kaiser-Meyer-Olkin (KMO) measure of sampling adequacy
  - KMO is a measure of the homogeneity of variables.
  - 0 <= KMO <= 1
  - Rule of thumb:

KMO	Interpretation
>.90	Very good
.80-.90	Good
.70-.80	OK
.60-.70	Acceptable
< .50	Unacceptable (no need for EFA)

Example 1

- Random data with no structure

```
my.df1 <- read.csv("EFA01.csv")
head(my.df1)
```

```
##         x1       x2       x3       x4       x5       x6
## 1 4.001819 6.169610 4.758423 5.111223 4.223895 4.728636
## 2 3.412486 5.004283 4.324779 1.791254 6.271531 6.556882
## 3 4.756883 6.014680 5.099372 5.729286 4.530885 4.174492
## 4 6.955701 4.296180 3.874160 5.547918 4.298545 5.210731
## 5 5.594236 6.085448 4.831808 5.776206 4.171769 5.404625
## 6 5.363289 3.283328 4.639084 5.510330 6.466561 3.738918
```

```
library(psych)
```

```
##
## Attaching package: 'psych'

## The following object is masked from 'package:lavaan':
##
##     cor2cov
```

```
KMO(my.df1)
```

```
## Kaiser-Meyer-Olkin factor adequacy
## Call: KMO(r = my.df1)
## Overall MSA =  0.36
## MSA for each item =
##   x1   x2   x3   x4   x5   x6
## 0.48 0.51 0.38 0.38 0.25 0.25
```

```
bartlett.test(my.df1)
```

```
##
##  Bartlett test of homogeneity of variances
##
## data:  my.df1
## Bartlett's K-squared = 4.6008, df = 5, p-value = 0.4665
```

```
cortest.bartlett(cor(my.df1),nrow(my.df1))
```

```
## $chisq
## [1] 27.51699
##
## $p.value
## [1] 0.02479569
##
## $df
## [1] 15
```

- KMO is not acceptable
- Bartlett's test is significant

Example 2

- Data with structure

```
my.df2 <- read.csv("EFA02.csv")

KMO(my.df2)
```

```
## Kaiser-Meyer-Olkin factor adequacy
## Call: KMO(r = my.df2)
## Overall MSA =  0.89
## MSA for each item =
##   x1   x2   x3   x4   x5   x6
## 0.90 0.88 0.89 0.87 0.90 0.90
```

```
cortest.bartlett(my.df2)
```

```
## R was not square, finding R from data
```

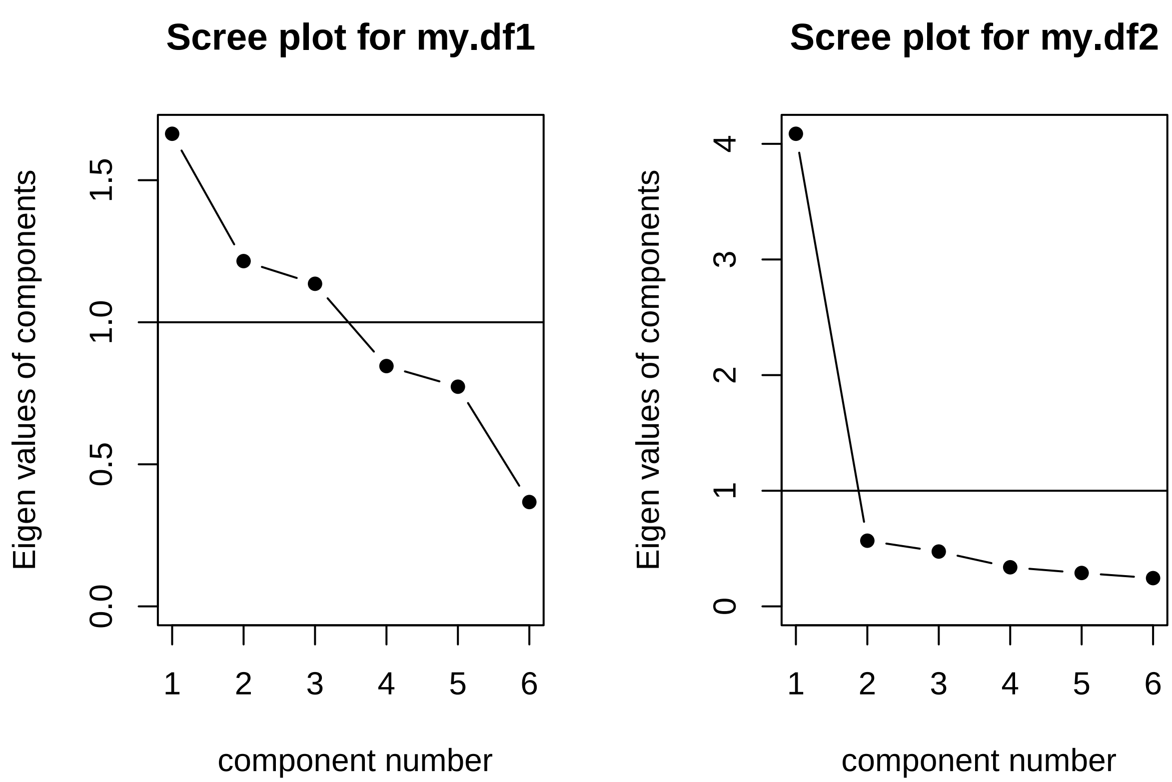
```
## $chisq
## [1] 168.0944
##
## $p.value
## [1] 5.895541e-28
##
## $df
## [1] 15
```

Extraction of initial factors

- We need to determine the no. of factors. There is no simple rule to determine it.
- The eigenvalue is a measure of how much of the variance of the observed variables a factor explains. Any factor with an eigenvalue 1 explains more variance than a single observed variable.
- Kaiser's (1960) criterion:
  - Eigenvalue/no. of variables = percentage of variance explained by the factor
  - They are in descending order. That is, the first factor explains most variance while the last one explains least.
  - Retain the factors with eigenvalues greater than 1.0. Why 1.0?
- Cattell's (1966) scree plot:

- Keep the factors in the steep slope before the first one which starts to level off.
- We should try several numbers of factors and see whether or not the extracted factors are interpretable (it is important in psychology!)

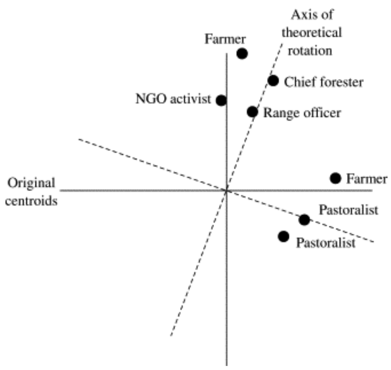
```
par(mfrow=c(1,2))
scree(my.df1, fa=FALSE, main="Scree plot for my.df1")
scree(my.df2, fa=FALSE, main="Scree plot for my.df2")
```



Factor rotation and interpretation

- Initial factor solutions (except the one-factor solutions) are usually uninterpretable.
- Because of factor indeterminacy, we may transform the initial factor solution into a simple structure.
- Simple structure:
  - Each variable is loaded into a few factors, preferably one;
  - Each factor is only related to a few variables.
- Orthogonal rotation (varimax in the psych package): the rotated factors are uncorrelated with each other.
- Oblique rotation (oblimin in the psych package): the rotated factors can be correlated with each other.

- Orthogonal or oblique rotations?
  - Orthogonal: With strong theoretical (or empirical) reasons that the factors are uncorrelated
  - Oblique: More realistic
  - Note. Results without rotation are likely incorrect.



Question: Is the overall variances explained changed before and after the rotation?

Interpretations of factors

- Items with high loadings (absolute value >.3 or .4) are grouped together.
- Based on the items, we may label the factors accordingly
- After rotations:
  - Pattern matrix: similar to regression coefficients,
    * e.g., $\hat{x}_1 = 0.65 * f_1 + 0.15 * f_2$.
  - Structure matrix: simple correlation between the items and the factors,
    * e.g., $Cor(x1, f1)$.

Practice

```
(fa1 <- fa(my.df1, nfactors=1, fm="ml"))
```

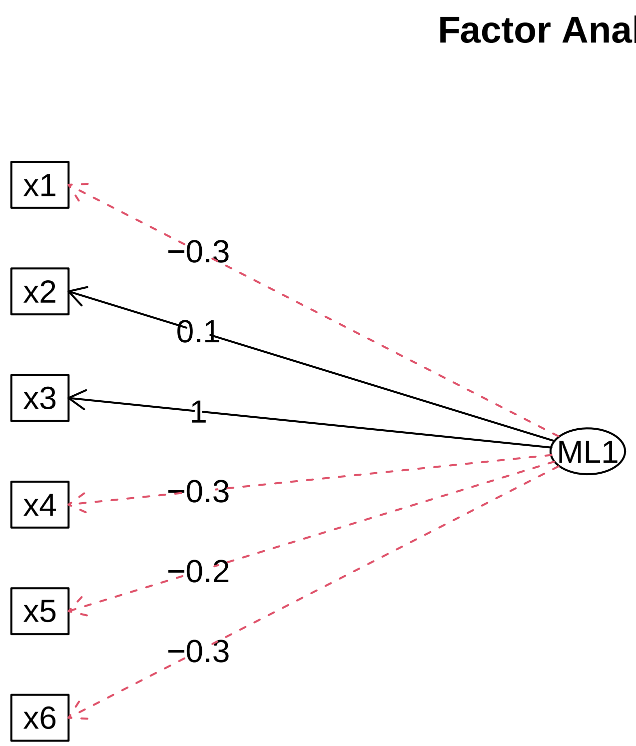
Example 1: Random data

```
## Factor Analysis using method =  ml
## Call: fa(r = my.df1, nfactors = 1, fm = "ml")
## Standardized loadings (pattern matrix) based upon correlation matrix
##       ML1    h2    u2 com
## x1 -0.28 0.078 0.922   1
## x2  0.14 0.021 0.979   1
```

```
## x3  1.00 0.995 0.005   1
## x4 -0.27 0.075 0.925   1
## x5 -0.17 0.029 0.971   1
## x6 -0.27 0.074 0.926   1
##
##                 ML1
## SS loadings    1.27
## Proportion Var 0.21
##
## Mean item complexity =  1
## Test of the hypothesis that 1 factor is sufficient.
##
## The degrees of freedom for the null model are  15  and the objective function was  0.6 with Chi Squa
## The degrees of freedom for the model are 9  and the objective function was  0.31
##
## The root mean square of the residuals (RMSR) is  0.13
## The df corrected root mean square of the residuals is  0.16
##
## The harmonic number of observations is  50 with the empirical chi square  23.53  with prob <  0.0051
## The total number of observations was  50  with Likelihood Chi Square =  14.15  with prob <  0.12
##
## Tucker Lewis Index of factoring reliability =  0.292
## RMSEA index =  0.105  and the 90 % confidence intervals are  0 0.21
## BIC =  -21.06
## Fit based upon off diagonal values = 0.52
## Measures of factor score adequacy
##                                                    ML1
## Correlation of (regression) scores with factors   1.00
## Multiple R square of scores with factors          1.00
## Minimum correlation of possible factor scores     0.99
```

- There is a factor loading of 1!

```
## cut=0: show all values
fa.diagram(fa1, cut=0, sort=FALSE)
```



```r
( fa2 <- fa(my.df2, nfactors=1, fm="ml") )
```

Example 2: 1 factor model

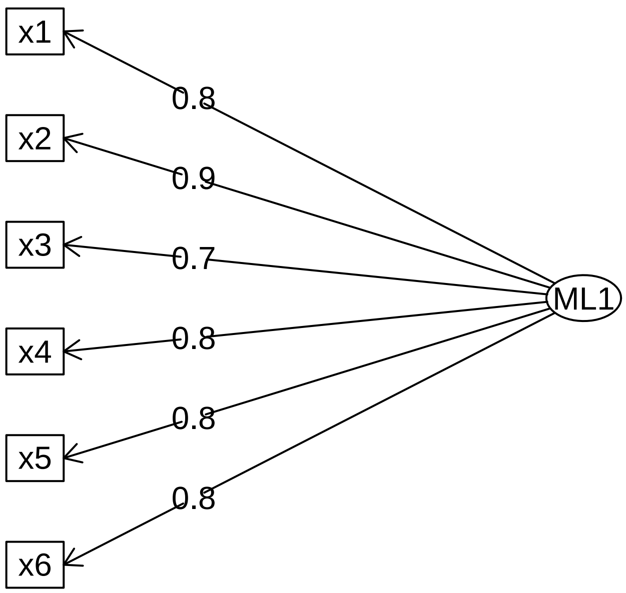
```
## Factor Analysis using method =  ml
## Call: fa(r = my.df2, nfactors = 1, fm = "ml")
## Standardized loadings (pattern matrix) based upon correlation matrix
##     ML1   h2   u2 com
## x1 0.79 0.62 0.38   1
## x2 0.87 0.75 0.25   1
## x3 0.70 0.49 0.51   1
## x4 0.75 0.57 0.43   1
## x5 0.82 0.68 0.32   1
## x6 0.79 0.62 0.38   1
##
##                 ML1
## SS loadings    3.72
## Proportion Var 0.62
##
## Mean item complexity =  1
```

```
## Test of the hypothesis that 1 factor is sufficient.
##
## The degrees of freedom for the null model are  15  and the objective function was  3.64 with Chi Squa
## The degrees of freedom for the model are 9  and the objective function was  0.14
##
## The root mean square of the residuals (RMSR) is  0.04
## The df corrected root mean square of the residuals is  0.05
##
## The harmonic number of observations is  50 with the empirical chi square  2.11  with prob <  0.99
## The total number of observations was  50  with Likelihood Chi Square =  6.45  with prob <  0.69
##
## Tucker Lewis Index of factoring reliability =  1.028
## RMSEA index =  0  and the 90 % confidence intervals are  0 0.125
## BIC =  -28.76
## Fit based upon off diagonal values = 1
## Measures of factor score adequacy
##                                                    ML1
## Correlation of (regression) scores with factors   0.96
## Multiple R square of scores with factors          0.91
## Minimum correlation of possible factor scores     0.83
```

- RMSEA=0. The proposed model fits the data very well.
- All communalities are high.

```
## cut=0: show all values
fa.diagram(fa2, cut=0, sort=FALSE)
```

Factor Analysis



Estimation of factor scores for further analysis

- A composite score reflecting the value of each factor may be used for further analysis
- Bartlett's method: $\hat{f} = X\Psi^{-1}\Lambda(\Lambda^T\Psi^{-1}\Lambda)^{-1}$
  - All variables are used in calculating the factor scores.
  - Advantage: More accurate in estimating the factor scores.
  - Disadvantage: Difficult to apply to other datasets.
- Factor-based or unit-loading method (Cattell, 1952)
  - Select salient items (e.g., loadings $> |0.4|$) for each factor.
  - The composite scores are created by averaging the scores on those items with zero or unit weight.
  - Advantage: Easy to construct and apply to other settings.
  - Disadvantage: Less accurate because all variables have the same weightings.

Example 3: 2 factor model

```r
KMO(my.df3)
```

```
## Kaiser-Meyer-Olkin factor adequacy
## Call: KMO(r = my.df3)
## Overall MSA =  0.93
## MSA for each item =
##   h1   h2   h3   h4   h5   h6   h7   h8   h9  h10  h11  h12  h13  h14  h15  h16
## 0.95 0.93 0.96 0.81 0.92 0.94 0.92 0.79 0.94 0.96 0.94 0.78 0.91 0.96 0.94 0.79
##  h17  h18  h19  h20
## 0.95 0.94 0.93 0.96
```

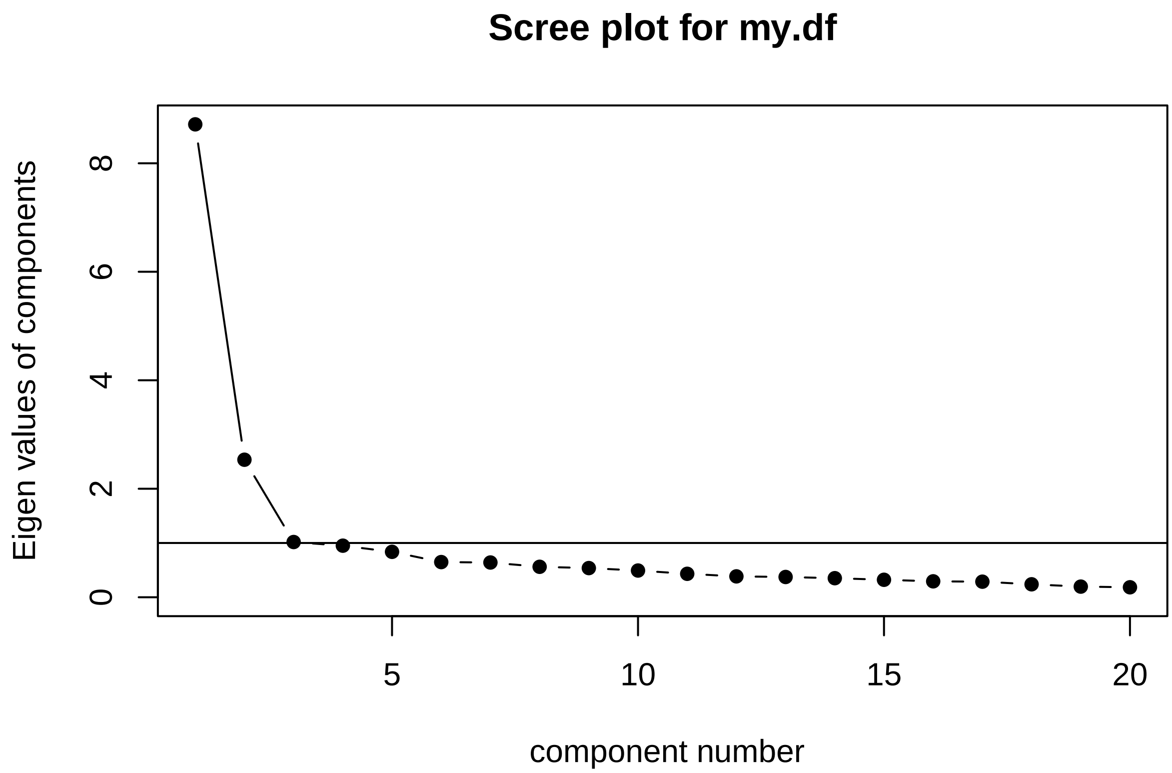
```r
cortest.bartlett(my.df3)
```

```
## R was not square, finding R from data
```

```
## $chisq
## [1] 3113.403
##
## $p.value
## [1] 0
##
## $df
## [1] 190
```

- It seems that both KMO and the Bartlett's test are okay.

```r
scree(my.df3, fa=FALSE, main="Scree plot for my.df")
```



- Both Kaiser's criterion and the scree test suggest 2 factors.
- Is the two-factor model interpretable?

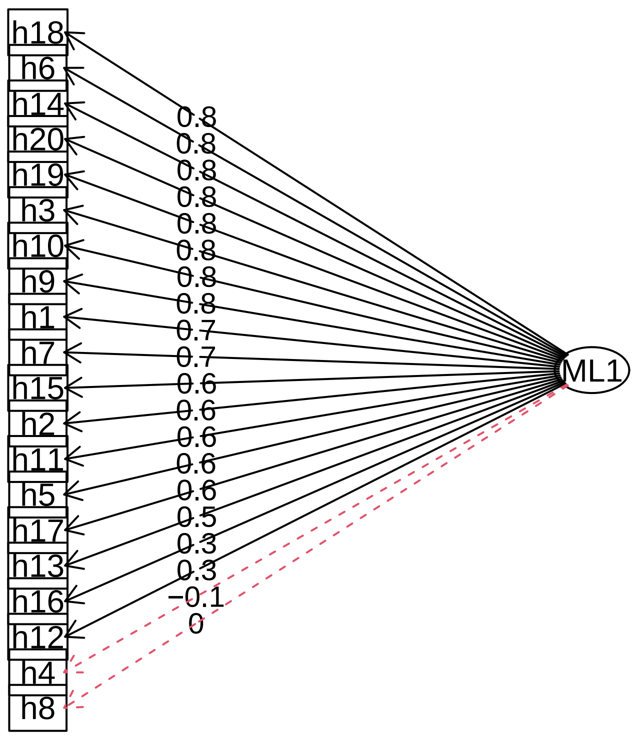
```
## One factor model
( fa3 <- fa(my.df3, nfactors=1, fm="ml") )
```

```
## Factor Analysis using method =  ml
## Call: fa(r = my.df3, nfactors = 1, fm = "ml")
## Standardized loadings (pattern matrix) based upon correlation matrix
##       ML1    h2   u2 com
## h1   0.66 0.437 0.56   1
## h2   0.61 0.376 0.62   1
## h3   0.78 0.616 0.38   1
## h4  -0.13 0.016 0.98   1
## h5   0.59 0.348 0.65   1
## h6   0.81 0.656 0.34   1
## h7   0.66 0.433 0.57   1
## h8  -0.03 0.001 1.00   1
## h9   0.75 0.563 0.44   1
## h10  0.78 0.612 0.39   1
## h11  0.60 0.355 0.65   1
## h12  0.32 0.101 0.90   1
## h13  0.48 0.230 0.77   1
## h14  0.80 0.644 0.36   1
```

```
## h15  0.65 0.418 0.58   1
## h16  0.34 0.114 0.89   1
## h17  0.59 0.346 0.65   1
## h18  0.84 0.698 0.30   1
## h19  0.79 0.620 0.38   1
## h20  0.80 0.640 0.36   1
##
##                  ML1
## SS loadings    8.22
## Proportion Var 0.41
##
## Mean item complexity =  1
## Test of the hypothesis that 1 factor is sufficient.
##
## The degrees of freedom for the null model are  190  and the objective function was  12.09 with Chi Sc
## The degrees of freedom for the model are 170  and the objective function was  3.11
##
## The root mean square of the residuals (RMSR) is  0.1
## The df corrected root mean square of the residuals is  0.11
##
## The harmonic number of observations is  266 with the empirical chi square  1113.8  with prob <  2.2e-
## The total number of observations was  266  with Likelihood Chi Square =  799.57  with prob <  3.2e-82
##
## Tucker Lewis Index of factoring reliability =  0.759
## RMSEA index =  0.118  and the 90 % confidence intervals are  0.11 0.127
## BIC =  -149.62
## Fit based upon off diagonal values = 0.94
## Measures of factor score adequacy
##                                                    ML1
## Correlation of (regression) scores with factors   0.97
## Multiple R square of scores with factors          0.95
## Minimum correlation of possible factor scores     0.90
```

```r
## cut=0: show all values
fa.diagram(fa3, cut=0)
```

Factor Analysis




```r
print( fa3$loadings, cut=0 )
```

```
##
## Loadings:
##      ML1
## h1    0.661
## h2    0.613
## h3    0.785
## h4   -0.125
## h5    0.590
## h6    0.810
## h7    0.658
## h8   -0.032
## h9    0.750
## h10   0.783
## h11   0.595
## h12   0.318
## h13   0.480
## h14   0.802
## h15   0.646
## h16   0.338
## h17   0.589
## h18   0.836
## h19   0.787
## h20   0.800
```

```
##
##                   ML1
## SS loadings    8.225
## Proportion Var 0.411
```

```
print( fa3$Structure, cut=0 )
```

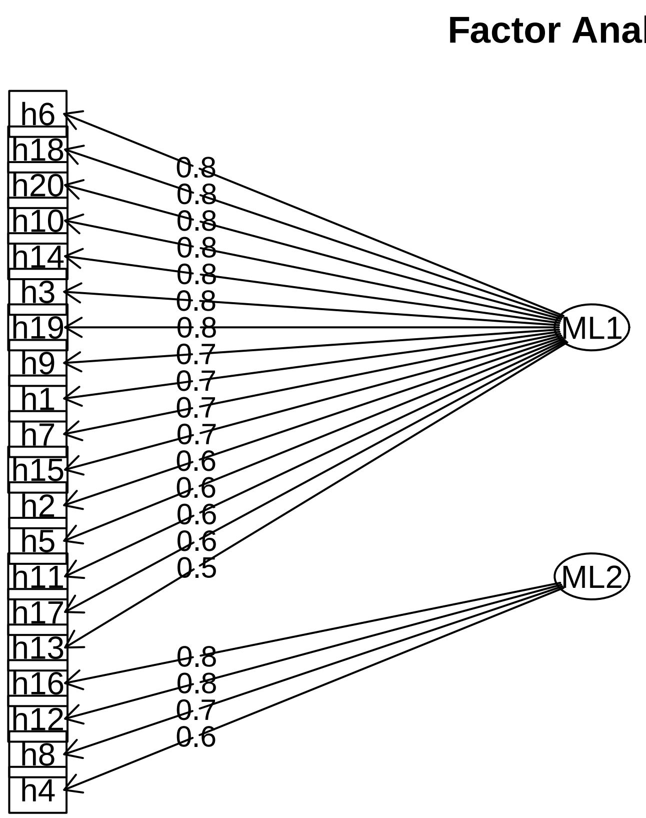
```
##
## Loadings:
##     ML1
## h1   0.661
## h2   0.613
## h3   0.785
## h4  -0.125
## h5   0.590
## h6   0.810
## h7   0.658
## h8  -0.032
## h9   0.750
## h10  0.783
## h11  0.595
## h12  0.318
## h13  0.480
## h14  0.802
## h15  0.646
## h16  0.338
## h17  0.589
## h18  0.836
## h19  0.787
## h20  0.800
##
##                   ML1
## SS loadings    8.225
## Proportion Var 0.411
```

```
## Two factor model
library(GPArotation)
( fa4 <- fa(my.df3, nfactors=2, fm="ml", rotate="oblimin") )
```

```
## Factor Analysis using method =  ml
## Call: fa(r = my.df3, nfactors = 2, rotate = "oblimin", fm = "ml")
## Standardized loadings (pattern matrix) based upon correlation matrix
##       ML1   ML2   h2   u2 com
## h1   0.71 -0.15 0.48 0.52 1.1
## h2   0.63 -0.08 0.39 0.61 1.0
## h3   0.78  0.03 0.62 0.38 1.0
## h4  -0.27  0.57 0.33 0.67 1.4
## h5   0.63 -0.16 0.39 0.61 1.1
## h6   0.80  0.02 0.66 0.34 1.0
## h7   0.69 -0.12 0.46 0.54 1.1
## h8  -0.20  0.69 0.46 0.54 1.2
## h9   0.71  0.15 0.57 0.43 1.1
## h10  0.79 -0.02 0.62 0.38 1.0
```

```
## h11  0.60 -0.03 0.36 0.64 1.0
## h12  0.14  0.79 0.69 0.31 1.1
## h13  0.49 -0.06 0.23 0.77 1.0
## h14  0.79  0.05 0.64 0.36 1.0
## h15  0.66 -0.05 0.43 0.57 1.0
## h16  0.16  0.79 0.70 0.30 1.1
## h17  0.57  0.07 0.35 0.65 1.0
## h18  0.80  0.14 0.70 0.30 1.1
## h19  0.77  0.08 0.62 0.38 1.0
## h20  0.80  0.01 0.64 0.36 1.0
##
##                        ML1  ML2
## SS loadings           8.16 2.16
## Proportion Var        0.41 0.11
## Cumulative Var        0.41 0.52
## Proportion Explained  0.79 0.21
## Cumulative Proportion 0.79 1.00
##
##  With factor correlations of
##     ML1 ML2
## ML1 1.0 0.2
## ML2 0.2 1.0
##
## Mean item complexity =  1.1
## Test of the hypothesis that 2 factors are sufficient.
##
## The degrees of freedom for the null model are  190  and the objective function was  12.09 with Chi Sc
## The degrees of freedom for the model are 151  and the objective function was  1.58
##
## The root mean square of the residuals (RMSR) is  0.05
## The df corrected root mean square of the residuals is  0.05
##
## The harmonic number of observations is  266 with the empirical chi square  218.72  with prob <  0.000
## The total number of observations was  266  with Likelihood Chi Square =  404.51  with prob <  4.9e-25
##
## Tucker Lewis Index of factoring reliability =  0.89
## RMSEA index =  0.079  and the 90 % confidence intervals are  0.07 0.089
## BIC =  -438.6
## Fit based upon off diagonal values = 0.99
## Measures of factor score adequacy
##                                                    ML1  ML2
## Correlation of (regression) scores with factors   0.97 0.92
## Multiple R square of scores with factors          0.95 0.85
## Minimum correlation of possible factor scores     0.90 0.70
```

```
fa.diagram(fa4)
```




```
print( fa4$loadings, cut=0 )
```

```
##
## Loadings:
##      ML1    ML2
## h1   0.705 -0.154
## h2   0.634 -0.075
## h3   0.780  0.033
## h4  -0.270  0.566
## h5   0.634 -0.156
## h6   0.805  0.024
## h7   0.693 -0.118
## h8  -0.204  0.692
## h9   0.713  0.150
## h10  0.788 -0.021
## h11  0.604 -0.026
## h12  0.136  0.790
## h13  0.492 -0.056
## h14  0.788  0.052
## h15  0.661 -0.051
## h16  0.157  0.791
## h17  0.570  0.067
## h18  0.800  0.138
## h19  0.765  0.077
## h20  0.799  0.007
```

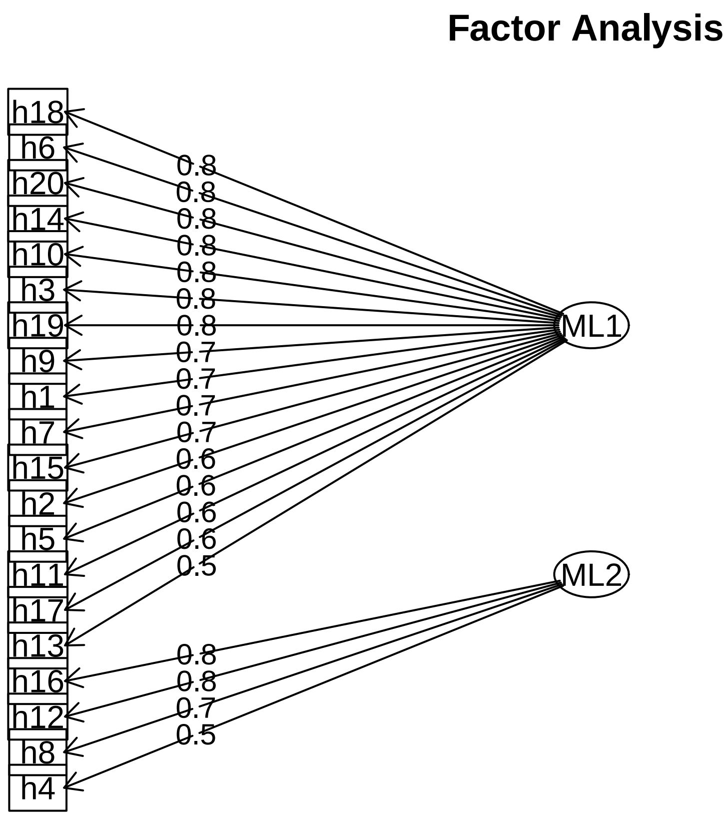
```
##
##                   ML1   ML2
## SS loadings    8.179 2.181
## Proportion Var 0.409 0.109
## Cumulative Var 0.409 0.518
```

```r
## Orthogonal ratation
( fa5 <- fa(my.df3, nfactors=2, fm="ml", rotate="varimax") )
```

```
## Factor Analysis using method =  ml
## Call: fa(r = my.df3, nfactors = 2, rotate = "varimax", fm = "ml")
## Standardized loadings (pattern matrix) based upon correlation matrix
##       ML1   ML2   h2   u2 com
## h1   0.68 -0.11 0.48 0.52 1.1
## h2   0.62 -0.04 0.39 0.61 1.0
## h3   0.78  0.08 0.62 0.38 1.0
## h4  -0.19  0.55 0.33 0.67 1.2
## h5   0.61 -0.12 0.39 0.61 1.1
## h6   0.81  0.07 0.66 0.34 1.0
## h7   0.67 -0.08 0.46 0.54 1.0
## h8  -0.10  0.67 0.46 0.54 1.0
## h9   0.73  0.19 0.57 0.43 1.1
## h10  0.78  0.02 0.62 0.38 1.0
## h11  0.60  0.01 0.36 0.64 1.0
## h12  0.25  0.79 0.69 0.31 1.2
## h13  0.48 -0.03 0.23 0.77 1.0
## h14  0.79  0.09 0.64 0.36 1.0
## h15  0.65 -0.01 0.43 0.57 1.0
## h16  0.27  0.79 0.70 0.30 1.2
## h17  0.58  0.10 0.35 0.65 1.1
## h18  0.82  0.18 0.70 0.30 1.1
## h19  0.78  0.12 0.62 0.38 1.0
## h20  0.80  0.05 0.64 0.36 1.0
##
##                        ML1  ML2
## SS loadings           8.18 2.15
## Proportion Var        0.41 0.11
## Cumulative Var        0.41 0.52
## Proportion Explained  0.79 0.21
## Cumulative Proportion 0.79 1.00
##
## Mean item complexity =  1.1
## Test of the hypothesis that 2 factors are sufficient.
##
## The degrees of freedom for the null model are  190  and the objective function was  12.09 with Chi So
## The degrees of freedom for the model are 151  and the objective function was  1.58
##
## The root mean square of the residuals (RMSR) is  0.05
## The df corrected root mean square of the residuals is  0.05
##
## The harmonic number of observations is  266 with the empirical chi square  218.72  with prob <  0.000
## The total number of observations was  266  with Likelihood Chi Square =  404.51  with prob <  4.9e-25
##
## Tucker Lewis Index of factoring reliability =  0.89
```

```
## RMSEA index =  0.079  and the 90 % confidence intervals are  0.07 0.089
## BIC =  -438.6
## Fit based upon off diagonal values = 0.99
## Measures of factor score adequacy
##                                                    ML1  ML2
## Correlation of (regression) scores with factors   0.97 0.92
## Multiple R square of scores with factors          0.95 0.84
## Minimum correlation of possible factor scores     0.90 0.69
```

```
fa.diagram(fa5)
```




```
print( fa5$loadings, cut=0 )
```

```
##
## Loadings:
##     ML1    ML2
## h1   0.682 -0.114
## h2   0.622 -0.040
## h3   0.784  0.075
## h4  -0.187  0.545
## h5   0.610 -0.119
## h6   0.807  0.068
## h7   0.674 -0.079
## h8  -0.102  0.674
## h9   0.734  0.187
```

```
## h10  0.784  0.023
## h11  0.600  0.008
## h12  0.251  0.789
## h13  0.483 -0.029
## h14  0.794  0.094
## h15  0.653 -0.014
## h16  0.272  0.791
## h17  0.579  0.098
## h18  0.819  0.181
## h19  0.775  0.118
## h20  0.798  0.050
##
##                  ML1   ML2
## SS loadings    8.175 2.150
## Proportion Var 0.409 0.107
## Cumulative Var 0.409 0.516
```

Computing factor scores

- We may group the items according to the factor structure:
  - F1= others
  - F2= h4+h8+h12+h16
- An alternative approach is to estimate the factor scores with Bartlett's method.

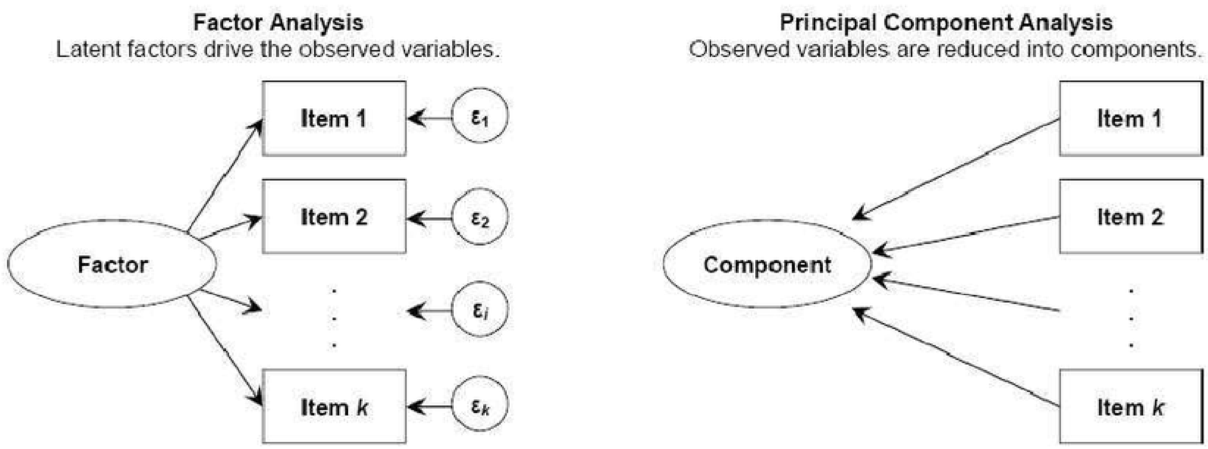
```
fs <- factor.scores(x=my.df3, f=fa4, method="Bartlett")
## Show the first few cases
head(fs$scores)
```

```
##             ML1        ML2
## [1,] -0.5036297 -1.0779198
## [2,] -0.2355948 -1.0983758
## [3,] -0.4158664  0.7726878
## [4,] -1.1289065 -0.5853815
## [5,]  2.3419891  0.9615771
## [6,] -1.3199167 -1.4453186
```

Principal component analysis (PCA) vs. factor analysis (FA)

- PCA:
  - $PC_1 = l_{11}x_1 + l_{12}x_2 + ... + l_{16}x_6$
  - $PC_2 = l_{21}x_1 + l_{22}x_2 + ... + l_{26}x_6$
  - All variables are observed.
  - There is no latent variable in PCA.

- FA:
  - $x_1 = \lambda_{11} f_1 + \lambda_{12} f_2 + e_1$
  - ...
  - $x_6 = \lambda_{61} f_1 + \lambda_{62} f_2 + e_6$
  - There are latent factors in FA.
- Research objectives:
  - PCA: Summarize information (variance) of the data.
  - FA: Identify underlying structures.
- Handling variances of the variables
  - PCA: all variances in the observed variables is analyzed
  - FA: error variance is estimated and removed from the analysis
- Practically, both results may be very similar.



Confirmatory factor analysis (CFA)

- Linear relationships among latent and observed variables.
- No direct effect among the latent variables.
- It is used to test the construct validity of psychological constructs.
- It is generally not advisable to apply CFA after EFA **on the same data set**.
- A new sample is required to replicate the findings.
- One possible strategy is to randomly split the data into two independent samples.
- Two common fallacies in factor analysis:
  - Naming fallacy: the belief that the factor is correctly labelled.
  - Reification fallacy: the belief that a hypothetical construct must correspond to a real thing.

Purposes of CFA

- Testing a single model:
  - Strictly confirmatory;
  - Reject or do not reject the proposed model.
- Comparing alternative models:
  - Compare several theoretically competing models;
  - The models can be nested or non-nested.
- Testing and modifying the model:
  - Reject or do not reject the proposed model;
  - Modify the proposed model if it is rejected.

Steps in CFA

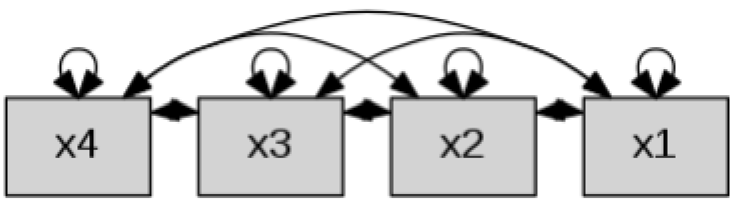
- Model specification: What is the proposed model?
- Model identification: Will there be any solutions for the model?
- Parameter estimation: What are the results for the model?
- Goodness of fit assessment: Does the model fit the data?
- Model modification and comparison: Which model is better? What are we going to do if the initial model does not fit the data?

Model specification

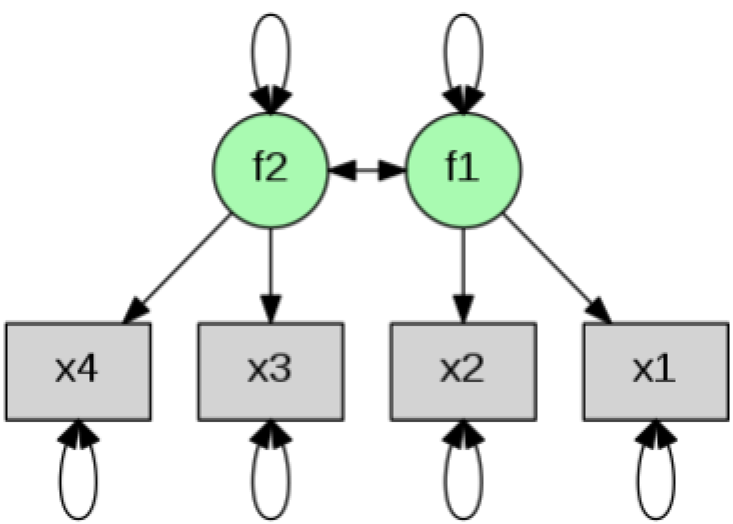
- What is the proposed model?
- $x = \mu_x + \Lambda f + e$
  - For example, $x_1 = \mu_{x_1} + \lambda_{11} f_1 + e_1$
  - x_1: observed continuous variable
  - μ_{x1}: intercept of the observed variable. It is usually set at the sample mean.
  - f_1: latent factor. Its mean is usually fixed at 0. The mean can be non-zero in multiple-group analysis and latent growth model.
  - $_{11}$: factor loading. It is similar to regression coefficient.
  - e_1: measurement error (and unique factor).

Local independence

- Measurement errors are usually assumed uncorrelated.
- This is known as the local independence assumption- residuals are uncorrelated after controlling for the latent factors.
- Measurement errors may be correlated in the presence of method variance, e.g., using positively and negatively wordings.
- Data:

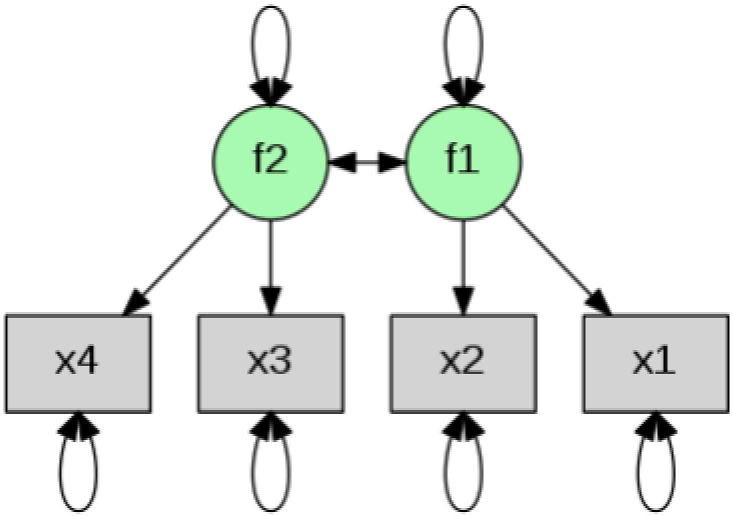


- Model:



Model implied matrices

- Factor loading matrix: Λ
- Factor covariance matrix: Φ
- Error variance matrix: Ψ
- Mean vector: $E(x) = \mu x + \Lambda E(f) + E(e) = \mu x + \Lambda E(f)$ as $E(e) = 0$
- Covariance matrix: $Cov(x) = \Lambda \Phi \Lambda T + \Psi$
- A two-factor model:



Is the model valid for testing?

- The t-rule:
  - Degrees of freedom (dfs) of the model: $p * -q$, where $p* = p(p + 1)/2$, p is the no. of variables, and q is the no. of free parameters.
  - A necessary but not sufficient condition for identification is that df is non-negative.
- A CFA model with one single factor:
  - A model is identified with three or more indicators.
- A CFA model with two or more factors is identified if (two-indicator rule, Bollen, 1989):
  - Factor correlations are free;
  - Two or more indicators per factor;
  - Each indicator loads on one factor; and
  - Errors are uncorrelated.
- Can we fit models with a factor with only 1 indicator?

Is the model identified?

- If there is no constraint, p*=10, q=11, df=-1.
- Metric of a latent variable:
  - What is the mean of a latent variable?
  - What is the variance of a latent variable?

Solutions

- To overcome the identification problem in our example, we have to:
- Fix the factor variances at some non-zero values, usually 1.0.
- This applies to independent (exogenous) latent variables only.
-
$$df = 1 : \Phi = \begin{bmatrix} 1.0 & \\ cor(f_2, f_1) & 1.0 \end{bmatrix} and\ \Lambda = \begin{bmatrix} \lambda_{11} & 0 \\ \lambda_{21} & 0 \\ 0 & \lambda_{32} \\ 0 & \lambda_{42} \end{bmatrix}$$
- We cannot test the significance of the factor variances as they are fixed parameters.

An alternative solution

- Fix a loading from the latent variable to one observed variable at some non-zero value, usually 1.0.
- This applies to both independent and dependent (endogenous) latent variables.
-
$$df = 1 : \Phi = \begin{bmatrix} var(f_1) & \\ cor(f_2, f_1) & var(f_2) \end{bmatrix} and\ \Lambda = \begin{bmatrix} 1 & 0 \\ \lambda_{21} & 0 \\ 0 & 1 \\ 0 & \lambda_{42} \end{bmatrix}$$
- We cannot test the significance of these two loadings.
- You need to fix one factor (loading) per latent factor.

Which approach should I use?

- They are usually equivalent in theory.
- Equivalent models: the model fit indices are exactly the same for them.
- However, we usually have preferences in fitting some models:
  - Single group analysis: Fixing the factor variances at 1.0
  - Multiple-group analysis: Fixing the loadings at 1.0
- Can we estimate both μ_x and μ_f in $x = \mu_x + \Lambda f + e$?

Parameter estimation

- Theory: $\Sigma = \Lambda\Phi\Sigma^T + \Psi$
- Reality: $S \approx \hat{\Lambda}\hat{\Phi}\hat{\Lambda}^T + \hat{\Psi} = \hat{\Sigma}$
- We try to find the values of the parameters such that the model implied covariance matrix is as similar to the sample covariance matrix as possible.

A CFA example

- Sample data: CFA01.csv, n =200
- Default options in lavaan and most SEM packages
  - One factor loading per factor is fixed at 1.0 and factor variances are free.
  - Values of the factor loadings are relative to the reference indicator.
  - No standard error and test statistic for fixed parameters.

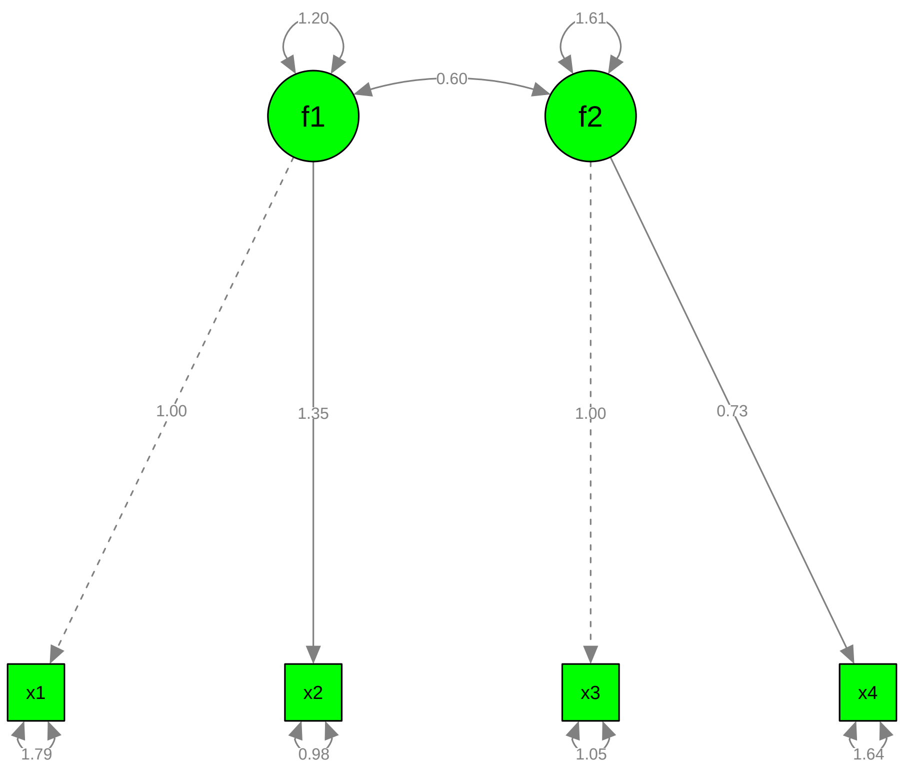
R code

```
library("lavaan")

my.df1 <- read.csv("CFA01.csv", header=TRUE)

## "=~" means f1 is measured by x1 and x2
my.model1 <- 'f1 =~ x1 + x2
f2 =~ x3 + x4'

my.fit1 <- cfa(my.model1, data=my.df1)
summary(my.fit1)
```

```
## lavaan 0.6-9 ended normally after 39 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                         9
##
##   Number of observations                           200
##
## Model Test User Model:
##
##   Test statistic                                 1.563
##   Degrees of freedom                                 1
##   P-value (Chi-square)                           0.211
##
## Parameter Estimates:
```

```
##
##   Standard errors                             Standard
##   Information                                 Expected
##   Information saturated (h1) model          Structured
##
## Latent Variables:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   f1 =~
##     x1                1.000
##     x2                1.354    0.378    3.579    0.000
##   f2 =~
##     x3                1.000
##     x4                0.729    0.223    3.274    0.001
##
## Covariances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   f1 ~~
##     f2                0.598    0.199    2.998    0.003
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##    .x1                1.794    0.367    4.888    0.000
##    .x2                0.978    0.595    1.644    0.100
##    .x3                1.051    0.486    2.161    0.031
##    .x4                1.645    0.301    5.465    0.000
##     f1                1.196    0.400    2.992    0.003
##     f2                1.613    0.534    3.019    0.003
```

```
semPaths(my.fit1, whatLabels="est", color="green")
```



Fix the factor variances for identification

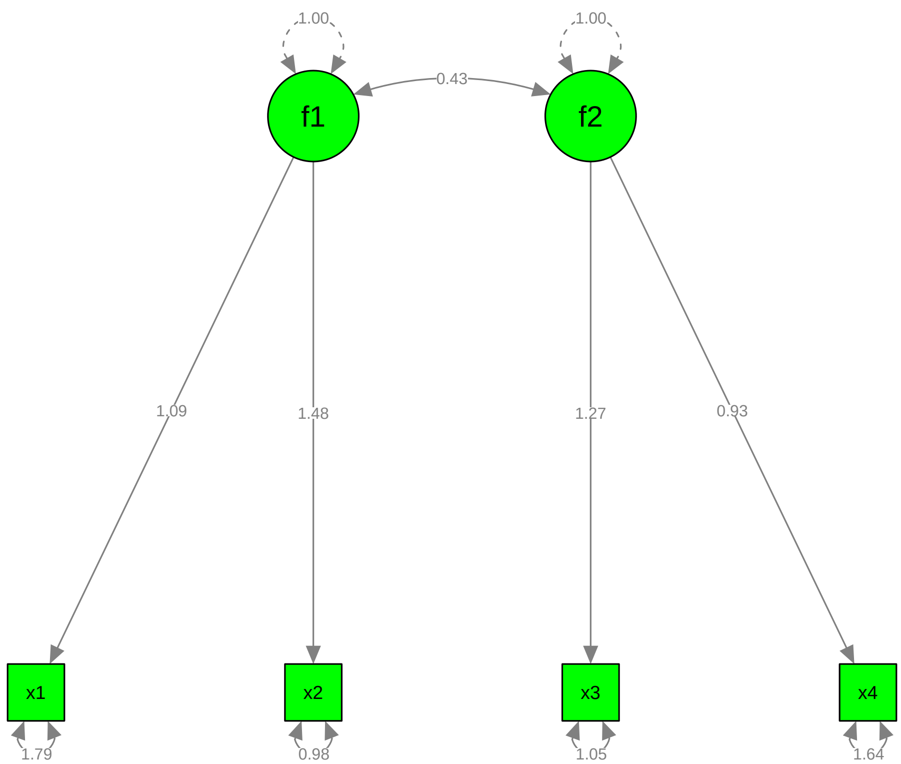
- It is possible (and preferable) to fix the factor variances for identification.
- Please note that the factor loadings can be larger than 1.

```
## f1 ~~ 1*f1 means that the variance of f1 is fixed at 1.
## f1 =~ NA*x1 means that the factor loading is unknown, i.e., free.
## f1 =~ start(1)*x2 means that the starting value is set at 1.
my.model2 <- 'f1 =~ NA*x1 + start(1)*x2
f2 =~ NA*x3 + x4
f1 ~~ 1*f1
f2 ~~ 1*f2'
my.fit2 <- cfa(my.model2, data=my.df1)

## Alternative model
## my.model2 <- 'f1 =~ x1 + x2
## f2 =~ x3 + x4'
## my.fit2 <- cfa(my.model2, data=my.df1, std.lv=TRUE)

summary(my.fit2)
```

```
## lavaan 0.6-9 ended normally after 28 iterations
```

```
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                         9
##
##   Number of observations                           200
##
## Model Test User Model:
##
##   Test statistic                                 1.563
##   Degrees of freedom                                 1
##   P-value (Chi-square)                           0.211
##
## Parameter Estimates:
##
##   Standard errors                             Standard
##   Information                                 Expected
##   Information saturated (h1) model          Structured
##
## Latent Variables:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   f1 =~
##     x1                1.094    0.183    5.985    0.000
##     x2                1.480    0.223    6.644    0.000
##   f2 =~
##     x3                1.270    0.210    6.037    0.000
##     x4                0.925    0.170    5.441    0.000
##
## Covariances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   f1 ~~
##     f2                0.431    0.099    4.344    0.000
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##     f1                1.000
##     f2                1.000
##    .x1                1.794    0.367    4.888    0.000
##    .x2                0.978    0.595    1.644    0.100
##    .x3                1.051    0.486    2.161    0.031
##    .x4                1.645    0.301    5.465    0.000
```

```
semPaths(my.fit2, whatLabels="est", color="green")
```



Standardized solutions

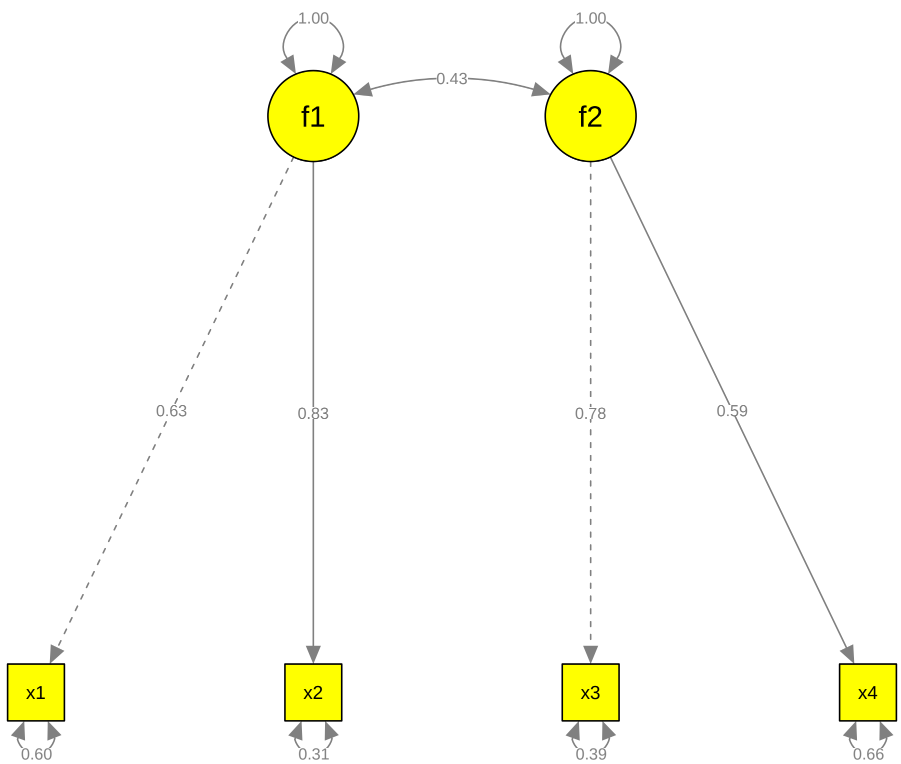
- Standardized solutions are used to study the relative values of the parameters.
- Standardized solutions can still be larger than 1 though this is not very likely (Joreskog, 1999).
- There are two forms of standardizations:
  - Standardization on the latent variables only (Std.lv in lavaan).
  - Standardization on both latent and observed variables (Std.all in lavaan).
- Standard errors are usually not reported for the standardized solutions.

```
## standardized=TRUE requests standardized solutions
summary(my.fit1, standardized=TRUE)
```

```
## lavaan 0.6-9 ended normally after 39 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                         9
##
##   Number of observations                           200
```

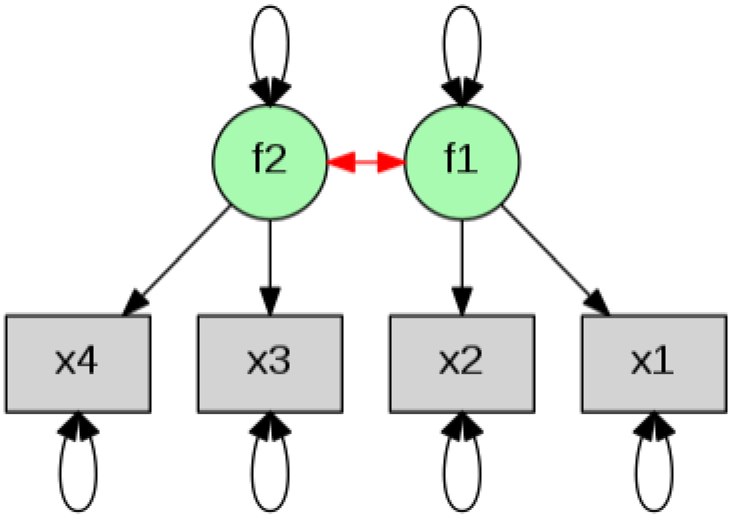
```
##
## Model Test User Model:
##
##   Test statistic                                 1.563
##   Degrees of freedom                                 1
##   P-value (Chi-square)                           0.211
##
## Parameter Estimates:
##
##   Standard errors                             Standard
##   Information                                 Expected
##   Information saturated (h1) model          Structured
##
## Latent Variables:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##   f1 =~
##     x1                1.000                               1.094    0.632
##     x2                1.354    0.378    3.579    0.000    1.480    0.832
##   f2 =~
##     x3                1.000                               1.270    0.778
##     x4                0.729    0.223    3.274    0.001    0.925    0.585
##
## Covariances:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##   f1 ~~
##     f2                0.598    0.199    2.998    0.003    0.431    0.431
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##    .x1                1.794    0.367    4.888    0.000    1.794    0.600
##    .x2                0.978    0.595    1.644    0.100    0.978    0.309
##    .x3                1.051    0.486    2.161    0.031    1.051    0.395
##    .x4                1.645    0.301    5.465    0.000    1.645    0.658
##     f1                1.196    0.400    2.992    0.003    1.000    1.000
##     f2                1.613    0.534    3.019    0.003    1.000    1.000
```

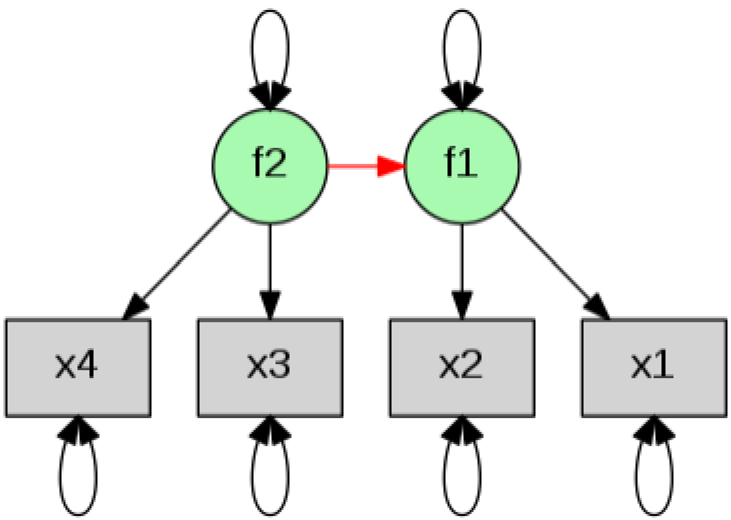
```r
## whatLabels="stand"
semPaths(my.fit1, whatLabels="stand", color="yellow")
```

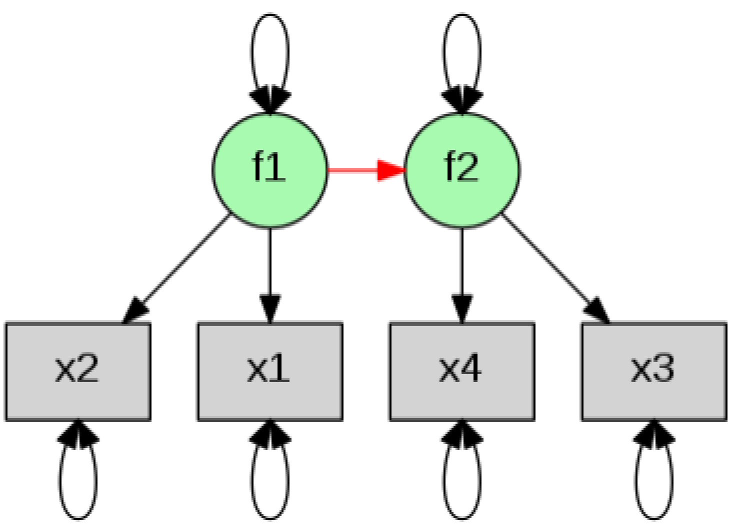


Equivalent models

- Models with the same model fit but with different interpretations (Raykov & Marcoulides, 2001).
- They have the same model implied mean vector and model implied covariance matrix, chi-square statistic, df and goodness-of-fit indices.
- They are all equally good/bad.
- The substantive meanings can be different.
- Correlated constructs vs. constructs with directions





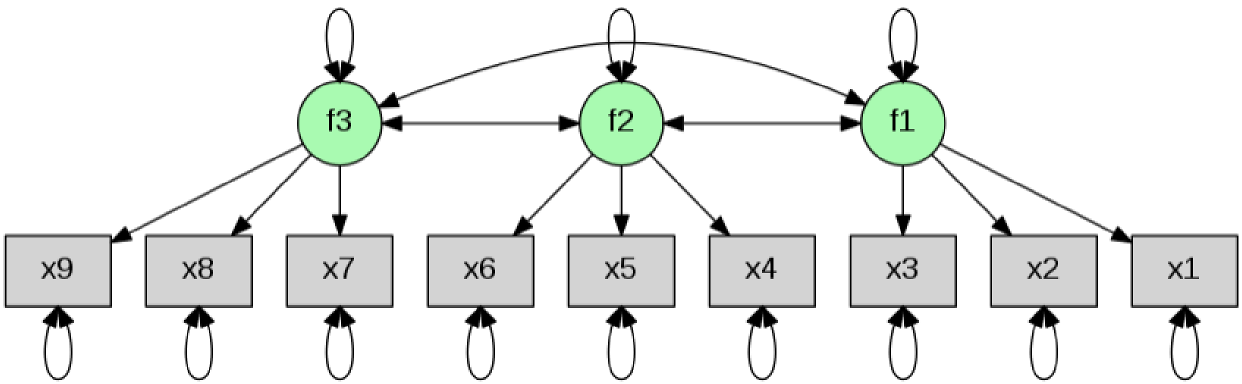


- It is nearly impossible to avoid equivalent models.
- It is difficult to generate all possible equivalent models.

- However, researchers should be aware of the alternative models.
- The best strategy is to build the model based on sound theories.

Other CFA models: Unidimensional measurement

- Each indicator loads on one factor only and the errors are uncorrelated.




```
my.df2 <- read.csv("CFA02.csv", header=TRUE)

my.model4 <- 'f1 =~ x1 + x2 + x3
f2 =~ x4 + x5 + x6
f3 =~ x7 + x8 + x9'

summary( my.fit4 <- cfa(my.model4, data=my.df2, std.lv=TRUE) )
```

```
## lavaan 0.6-9 ended normally after 26 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                        21
##
##   Number of observations                           500
##
## Model Test User Model:
##
##   Test statistic                                23.298
##   Degrees of freedom                                24
##   P-value (Chi-square)                           0.502
##
## Parameter Estimates:
##
##   Standard errors                             Standard
##   Information                                 Expected
##   Information saturated (h1) model          Structured
```

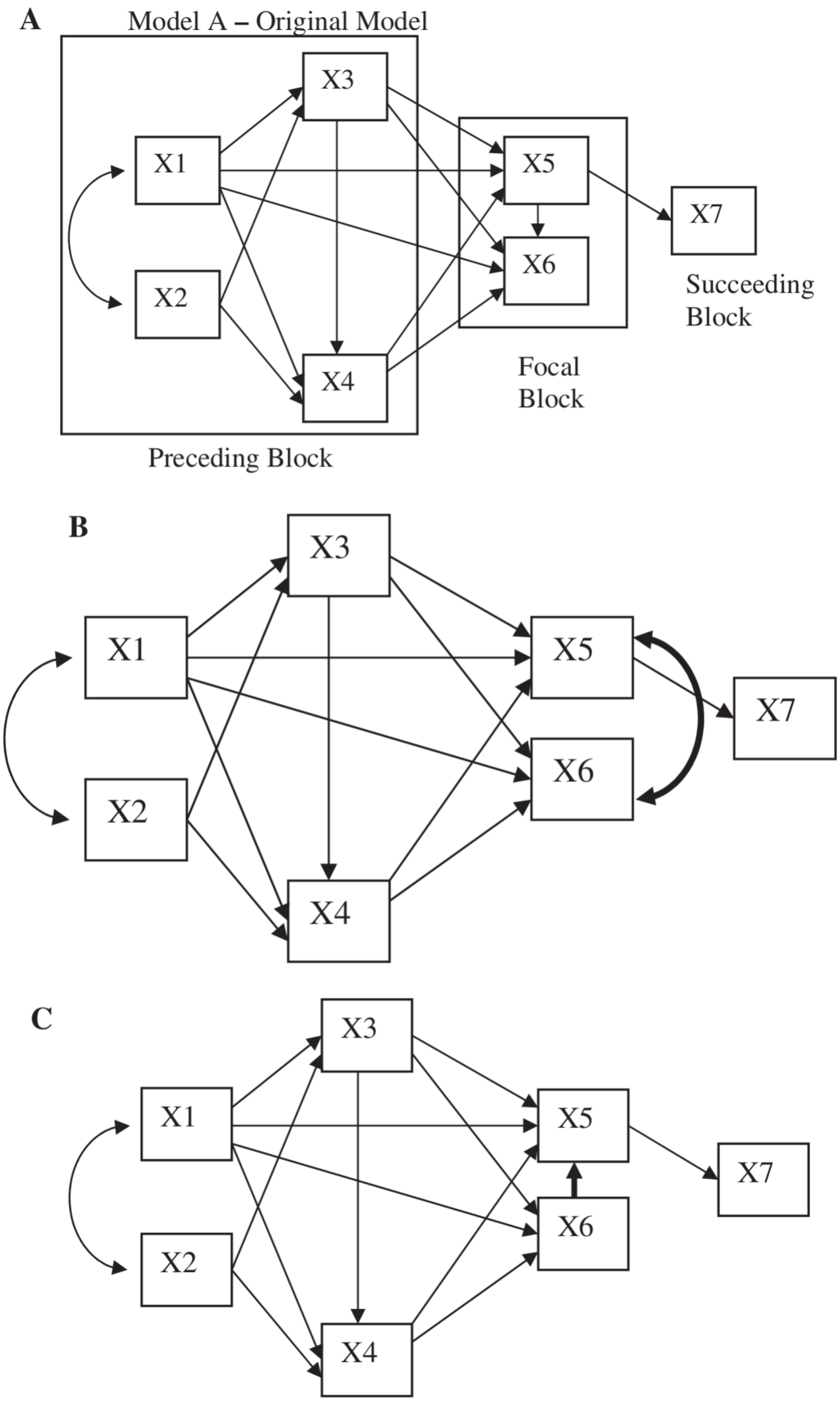


Figure 1: Equivalent models set 1

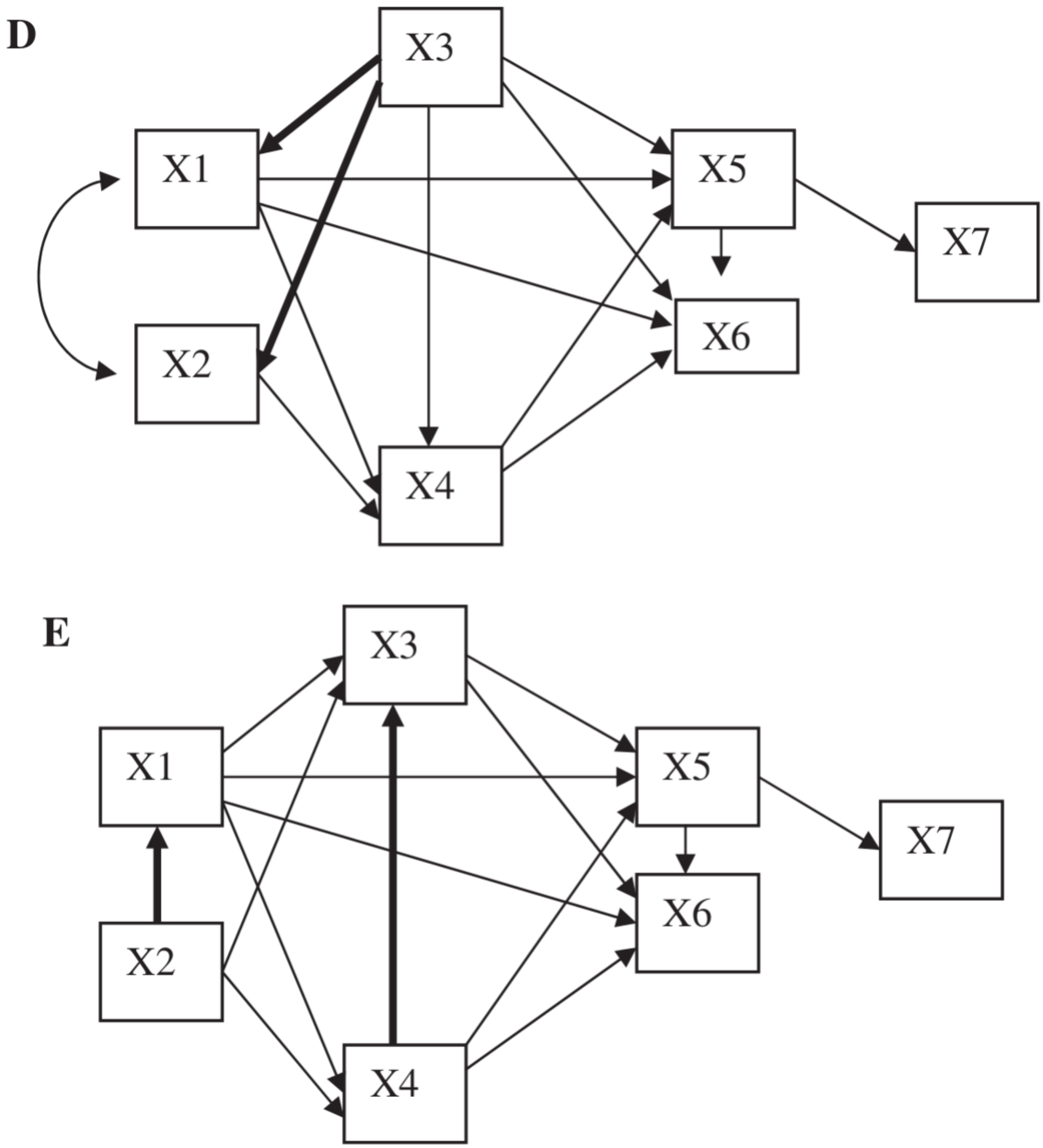
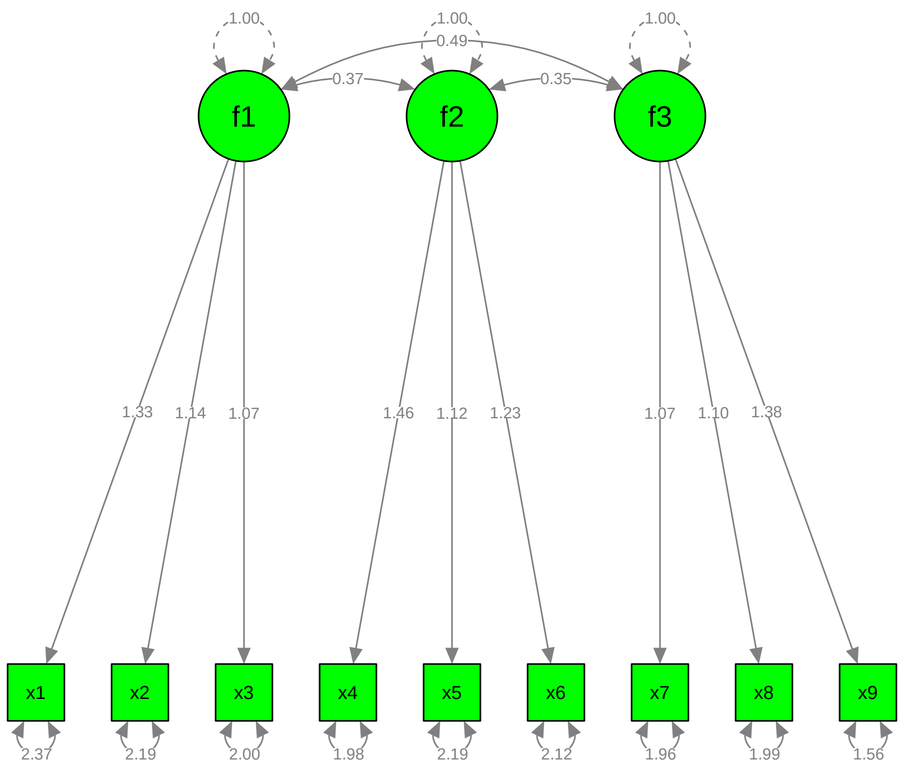


Figure 2: Equivalent models set 2

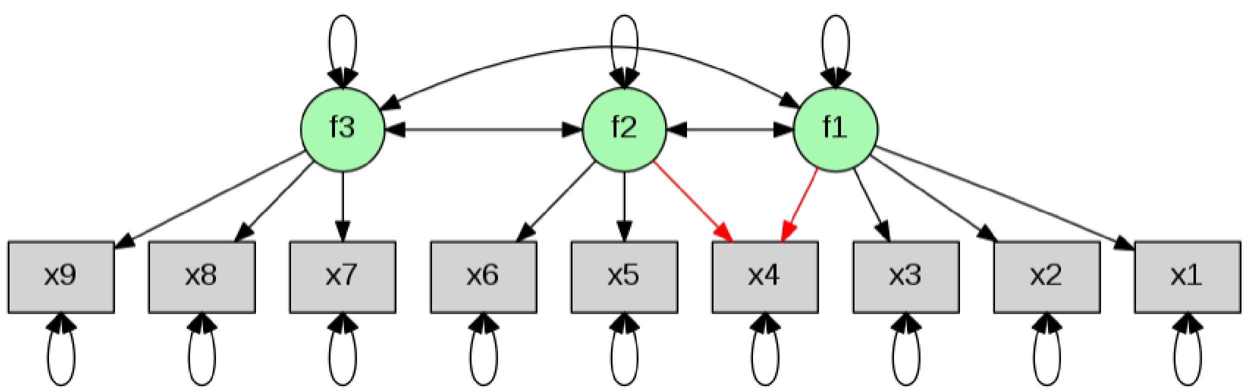
```
##
## Latent Variables:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   f1 =~
##     x1                1.326    0.106   12.569    0.000
##     x2                1.141    0.096   11.835    0.000
##     x3                1.066    0.091   11.677    0.000
##   f2 =~
##     x4                1.463    0.104   14.135    0.000
##     x5                1.116    0.093   12.059    0.000
##     x6                1.227    0.096   12.837    0.000
##   f3 =~
##     x7                1.073    0.087   12.338    0.000
##     x8                1.098    0.088   12.462    0.000
##     x9                1.385    0.093   14.843    0.000
##
## Covariances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   f1 ~~
##     f2                0.373    0.060    6.162    0.000
##     f3                0.489    0.057    8.647    0.000
##   f2 ~~
##     f3                0.347    0.059    5.921    0.000
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##    .x1                2.366    0.235   10.084    0.000
##    .x2                2.186    0.195   11.220    0.000
##    .x3                1.999    0.175   11.435    0.000
##    .x4                1.978    0.235    8.420    0.000
##    .x5                2.193    0.184   11.921    0.000
##    .x6                2.119    0.196   10.808    0.000
##    .x7                1.960    0.164   11.964    0.000
##    .x8                1.986    0.168   11.819    0.000
##    .x9                1.556    0.195    7.994    0.000
##     f1                1.000
##     f2                1.000
##     f3                1.000
```

```r
semPaths(my.fit4, whatLabels="est", color="green")
```



Multidimensional measurement

- An indicator loads on more than one factor

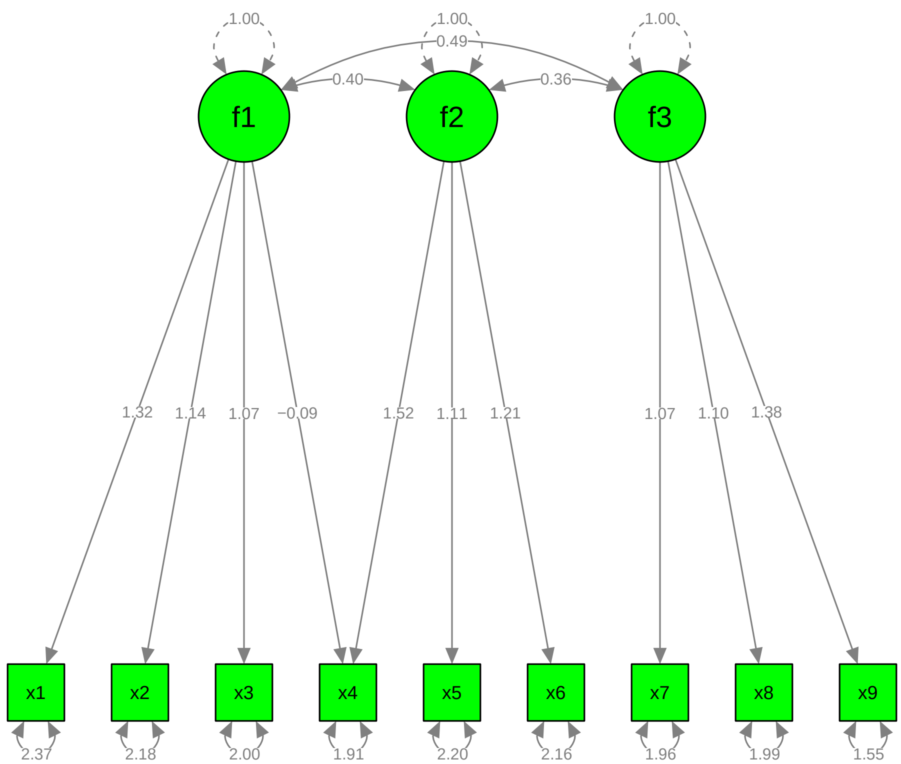



```
my.model5 <- 'f1 =~ x1 + x2 + x3 + x4
              f2 =~ x4 + x5 + x6
              f3 =~ x7 + x8 + x9'

summary( my.fit5 <- sem(my.model5, data=my.df2, std.lv=TRUE) )
```

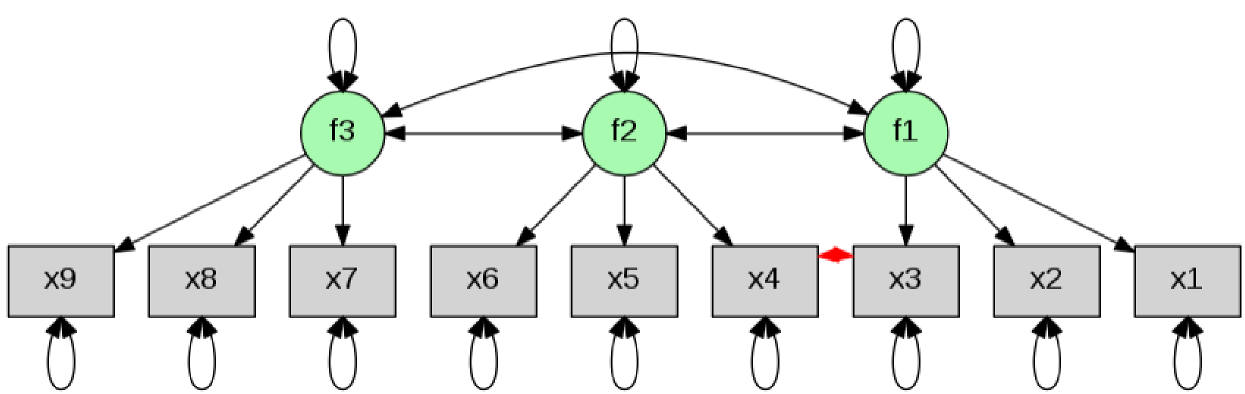
```
## lavaan 0.6-9 ended normally after 30 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                        22
##
##   Number of observations                           500
##
## Model Test User Model:
##
##   Test statistic                                22.805
##   Degrees of freedom                                23
##   P-value (Chi-square)                           0.472
##
## Parameter Estimates:
##
##   Standard errors                             Standard
##   Information                                 Expected
##   Information saturated (h1) model          Structured
##
## Latent Variables:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   f1 =~
##     x1                1.323    0.105   12.548    0.000
##     x2                1.142    0.096   11.846    0.000
##     x3                1.068    0.091   11.694    0.000
##     x4               -0.093    0.134   -0.691    0.489
##   f2 =~
##     x4                1.520    0.136   11.180    0.000
##     x5                1.112    0.093   11.940    0.000
##     x6                1.211    0.097   12.519    0.000
##   f3 =~
##     x7                1.073    0.087   12.335    0.000
##     x8                1.098    0.088   12.462    0.000
##     x9                1.385    0.093   14.847    0.000
##
## Covariances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   f1 ~~
##     f2                0.398    0.069    5.755    0.000
##     f3                0.490    0.056    8.680    0.000
##   f2 ~~
##     f3                0.356    0.060    5.960    0.000
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##    .x1                2.374    0.234   10.137    0.000
##    .x2                2.185    0.195   11.216    0.000
##    .x3                1.996    0.175   11.422    0.000
##    .x4                1.912    0.262    7.286    0.000
##    .x5                2.201    0.186   11.864    0.000
##    .x6                2.159    0.199   10.848    0.000
##    .x7                1.960    0.164   11.968    0.000
##    .x8                1.986    0.168   11.820    0.000
```

```
##    .x9                 1.555    0.195    7.991    0.000
##     f1                 1.000
##     f2                 1.000
##     f3                 1.000
```

```r
semPaths(my.fit5, whatLabels="est", color="green")
```



Correlated errors

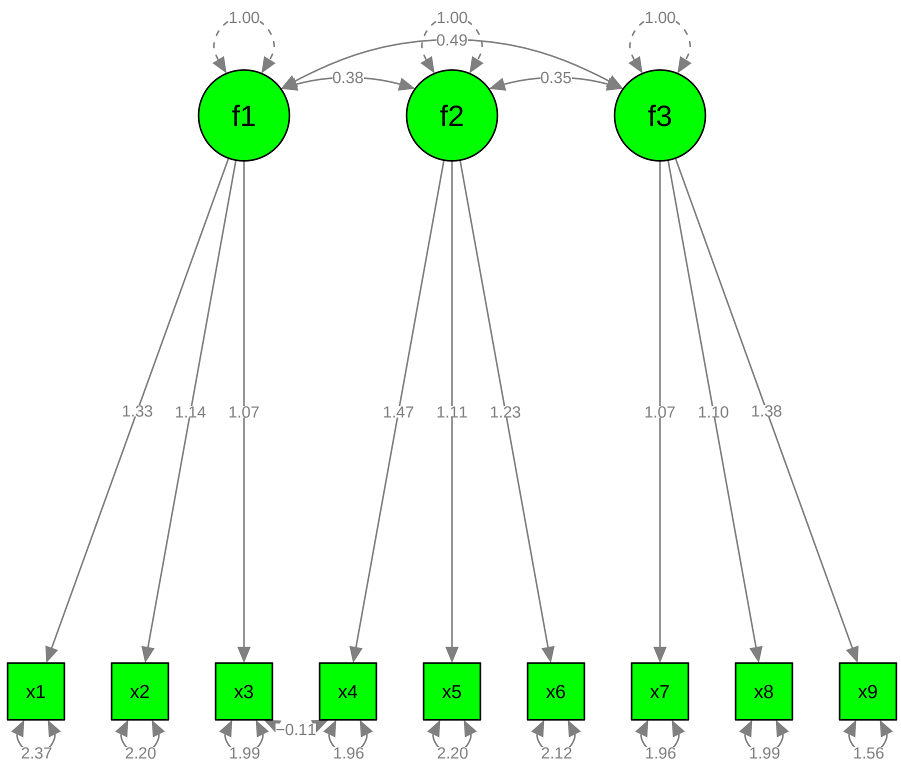


```r
my.model6 <- 'f1 =~ x1 + x2 + x3
              f2 =~ x4 + x5 + x6
              f3 =~ x7 + x8 + x9
              x3 ~~ x4 ## correlated errors'

summary( my.fit6 <- sem(my.model6, data=my.df2, std.lv=TRUE), standardized=TRUE )
```

```
## lavaan 0.6-9 ended normally after 27 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                        22
##
##   Number of observations                           500
##
## Model Test User Model:
##
##   Test statistic                                22.525
##   Degrees of freedom                                23
##   P-value (Chi-square)                           0.489
##
## Parameter Estimates:
##
##   Standard errors                             Standard
##   Information                                 Expected
##   Information saturated (h1) model          Structured
##
## Latent Variables:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##   f1 =~
##     x1                1.325    0.105   12.586    0.000    1.325    0.653
##     x2                1.137    0.096   11.817    0.000    1.137    0.609
##     x3                1.071    0.092   11.698    0.000    1.071    0.605
##   f2 =~
##     x4                1.469    0.104   14.158    0.000    1.469    0.724
##     x5                1.113    0.092   12.040    0.000    1.113    0.600
##     x6                1.225    0.095   12.829    0.000    1.225    0.643
##   f3 =~
##     x7                1.074    0.087   12.344    0.000    1.074    0.609
##     x8                1.099    0.088   12.466    0.000    1.099    0.615
##     x9                1.384    0.093   14.829    0.000    1.384    0.742
##
## Covariances:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##  .x3 ~~
##    .x4               -0.106    0.121   -0.880    0.379   -0.106   -0.054
##   f1 ~~
##     f2                0.382    0.061    6.228    0.000    0.382    0.382
##     f3                0.488    0.057    8.625    0.000    0.488    0.488
##   f2 ~~
##     f3                0.347    0.059    5.910    0.000    0.347    0.347
##
## Variances:
```

```
##                      Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##    .x1                  2.367    0.234   10.118    0.000    2.367    0.574
##    .x2                  2.196    0.194   11.306    0.000    2.196    0.630
##    .x3                  1.989    0.175   11.343    0.000    1.989    0.634
##    .x4                  1.962    0.236    8.309    0.000    1.962    0.476
##    .x5                  2.200    0.184   11.979    0.000    2.200    0.640
##    .x6                  2.124    0.196   10.856    0.000    2.124    0.586
##    .x7                  1.958    0.164   11.953    0.000    1.958    0.630
##    .x8                  1.985    0.168   11.809    0.000    1.985    0.622
##    .x9                  1.559    0.195    8.012    0.000    1.559    0.449
##     f1                  1.000                               1.000    1.000
##     f2                  1.000                               1.000    1.000
##     f3                  1.000                               1.000    1.000
```

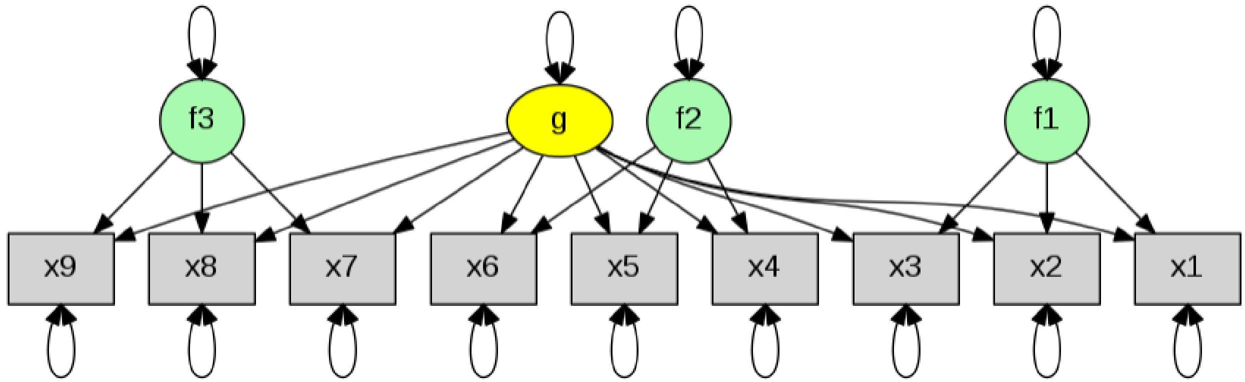
```
semPaths(my.fit6, whatLabels="est", color="green")
```



Higher-order CFA models

- Bi-factor or hierarchical CFA model (Rindskopf & Rose, 1988)
  - General factor with uncorrelated specific factors
  - For example, general intelligence vs. task specific skills

- There are usually more estimation problems in bi-factor models.

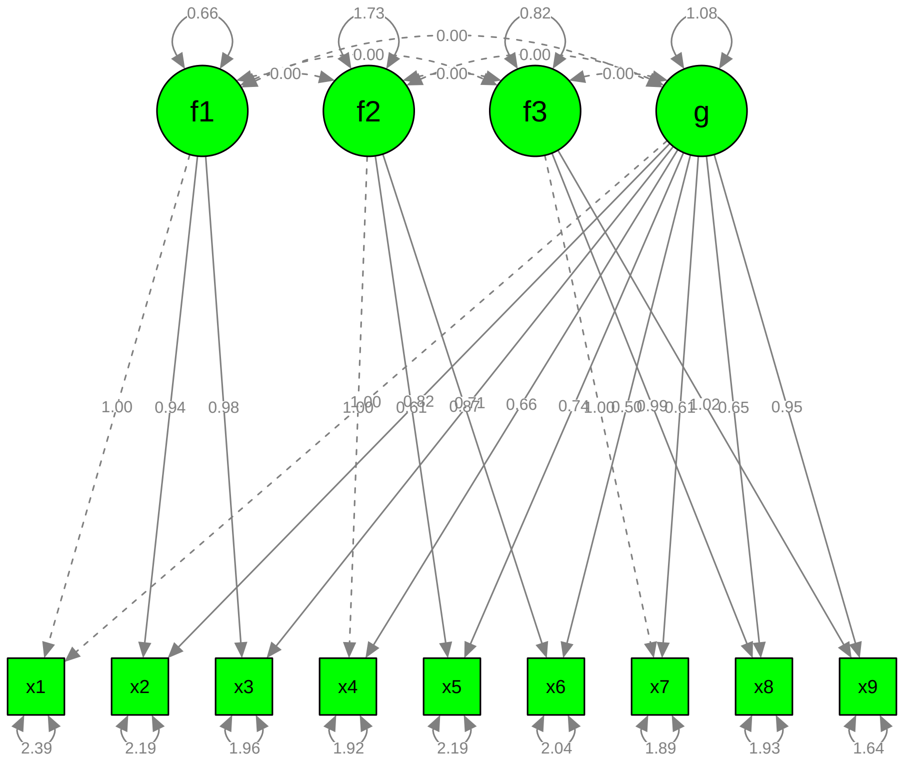



```r
## orthogonal=TRUE means that the latent factors are uncorrelated
my.model7 <- 'f1 =~ x1 + x2 + x3
f2 =~ x4 + x5 + x6
f3 =~ x7 + x8 + x9
g =~ x1 + x2 + x3 + x4 + x5 + x6 + x7 + x8 + x9'

summary( my.fit7 <- cfa(my.model7, orthogonal=TRUE, data=my.df2), standardized=TRUE )
```

```
## lavaan 0.6-9 ended normally after 81 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                        27
##
##   Number of observations                           500
##
## Model Test User Model:
##
##   Test statistic                                12.750
##   Degrees of freedom                                18
##   P-value (Chi-square)                           0.806
##
## Parameter Estimates:
##
##   Standard errors                             Standard
##   Information                                 Expected
##   Information saturated (h1) model          Structured
##
## Latent Variables:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##   f1 =~
##     x1                1.000                               0.811    0.399
##     x2                0.939    0.265    3.545    0.000    0.762    0.408
##     x3                0.978    0.296    3.302    0.001    0.794    0.448
##   f2 =~
##     x4                1.000                               1.316    0.648
##     x5                0.613    0.099    6.221    0.000    0.807    0.435
##     x6                0.871    0.146    5.979    0.000    1.147    0.602
```

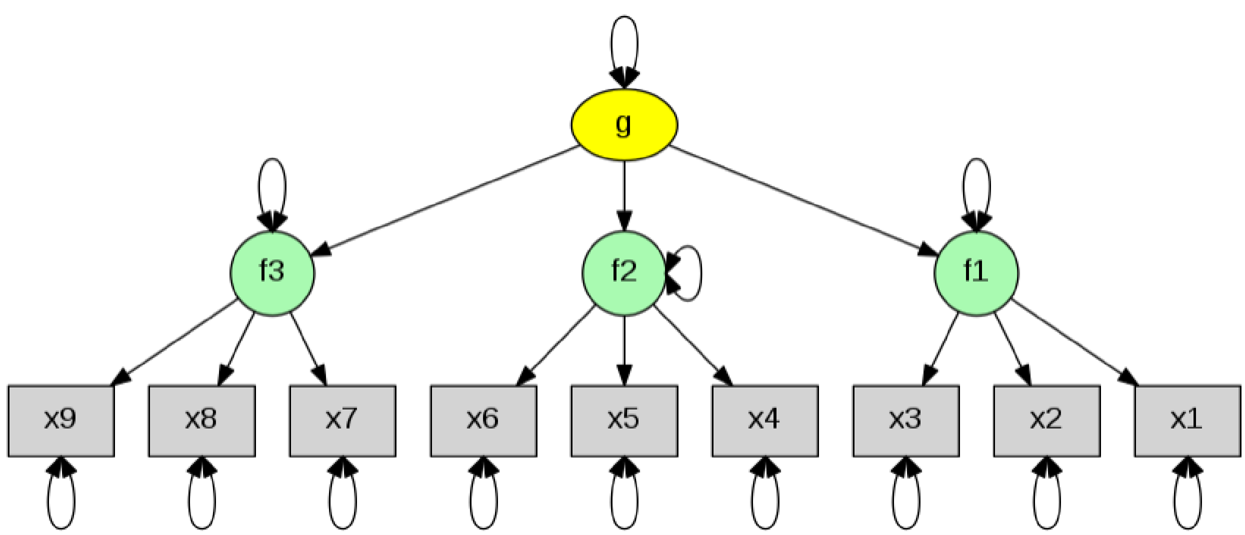
```
##   f3 =~
##     x7              1.000                               0.908    0.515
##     x8              0.991    0.197    5.029    0.000    0.900    0.504
##     x9              1.025    0.195    5.264    0.000    0.930    0.499
##   g =~
##     x1              1.000                               1.039    0.512
##     x2              0.815    0.130    6.294    0.000    0.847    0.453
##     x3              0.714    0.119    5.981    0.000    0.742    0.419
##     x4              0.658    0.156    4.216    0.000    0.684    0.337
##     x5              0.742    0.156    4.745    0.000    0.770    0.415
##     x6              0.499    0.136    3.661    0.000    0.519    0.273
##     x7              0.610    0.153    3.989    0.000    0.633    0.359
##     x8              0.647    0.158    4.087    0.000    0.672    0.376
##     x9              0.947    0.200    4.734    0.000    0.984    0.528
##
## Covariances:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##   f1 ~~
##     f2              0.000                               0.000    0.000
##     f3              0.000                               0.000    0.000
##     g               0.000                               0.000    0.000
##   f2 ~~
##     f3              0.000                               0.000    0.000
##     g               0.000                               0.000    0.000
##   f3 ~~
##     g               0.000                               0.000    0.000
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##    .x1              2.387    0.237   10.051    0.000    2.387    0.579
##    .x2              2.191    0.210   10.433    0.000    2.191    0.628
##    .x3              1.956    0.214    9.120    0.000    1.956    0.624
##    .x4              1.919    0.285    6.725    0.000    1.919    0.466
##    .x5              2.193    0.177   12.386    0.000    2.193    0.638
##    .x6              2.041    0.237    8.604    0.000    2.041    0.563
##    .x7              1.885    0.186   10.109    0.000    1.885    0.606
##    .x8              1.931    0.185   10.426    0.000    1.931    0.605
##    .x9              1.640    0.187    8.791    0.000    1.640    0.472
##     f1              0.658    0.304    2.162    0.031    1.000    1.000
##     f2              1.732    0.354    4.889    0.000    1.000    1.000
##     f3              0.824    0.230    3.589    0.000    1.000    1.000
##     g               1.079    0.298    3.625    0.000    1.000    1.000
```

```
semPaths(my.fit7, whatLabels="est", color="green")
```



Second-order CFA model

- Explaining the association of the first-order factors by higher-order factors.
- Identification is also an issue for the higher-order factors.

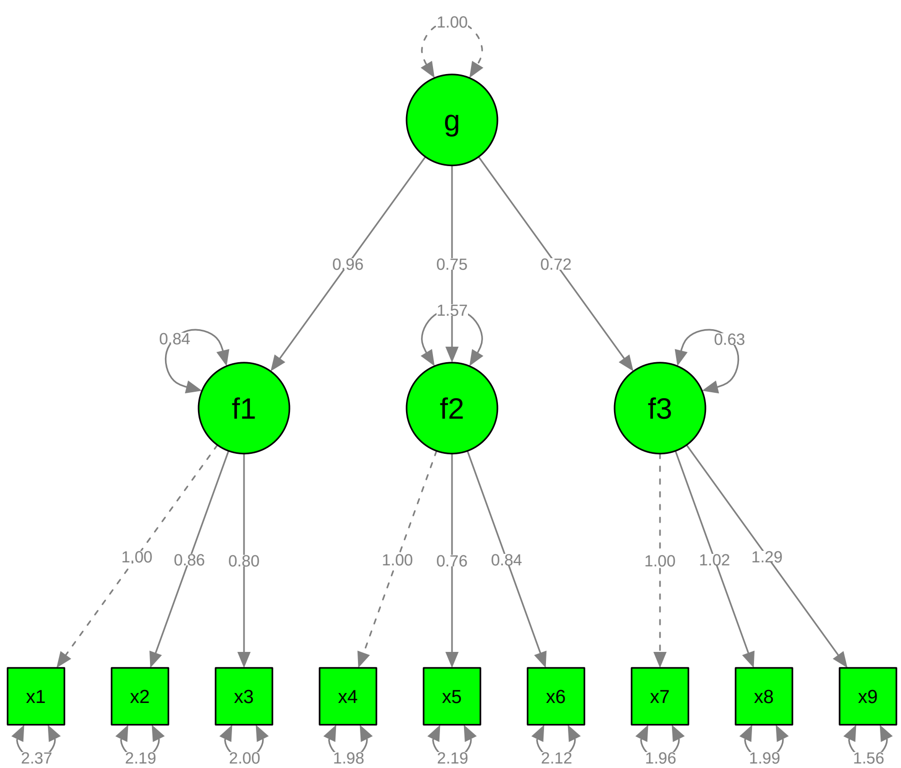


```r
my.model8 <- 'f1 =~ x1 + x2 + x3
f2 =~ x4 + x5 + x6
f3 =~ x7 + x8 + x9
g =~ NA*f1 + f2 + f3
g ~~ 1*g'

summary( my.fit8 <- sem(my.model8, data=my.df2), standardized=TRUE )
```

```
## lavaan 0.6-9 ended normally after 46 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                        21
##
##   Number of observations                           500
##
## Model Test User Model:
##
##   Test statistic                                23.298
##   Degrees of freedom                                24
##   P-value (Chi-square)                           0.502
##
## Parameter Estimates:
##
##   Standard errors                             Standard
##   Information                                 Expected
##   Information saturated (h1) model          Structured
##
## Latent Variables:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##   f1 =~
##     x1                1.000                               1.326    0.653
##     x2                0.860    0.099    8.712    0.000    1.141    0.611
##     x3                0.804    0.093    8.676    0.000    1.066    0.602
##   f2 =~
##     x4                1.000                               1.463    0.721
##     x5                0.763    0.082    9.337    0.000    1.116    0.602
##     x6                0.839    0.088    9.501    0.000    1.227    0.645
##   f3 =~
##     x7                1.000                               1.073    0.608
##     x8                1.023    0.109    9.432    0.000    1.098    0.615
##     x9                1.291    0.134    9.643    0.000    1.385    0.743
##   g =~
##     f1                0.960    0.133    7.210    0.000    0.724    0.724
##     f2                0.753    0.114    6.596    0.000    0.515    0.515
##     f3                0.724    0.105    6.920    0.000    0.675    0.675
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##     g                 1.000                               1.000    1.000
##    .x1                2.366    0.235   10.084    0.000    2.366    0.574
##    .x2                2.186    0.195   11.220    0.000    2.186    0.627
##    .x3                1.999    0.175   11.435    0.000    1.999    0.637
```

```
##    .x4               1.978    0.235    8.420    0.000    1.978    0.480
##    .x5               2.193    0.184   11.921    0.000    2.193    0.638
##    .x6               2.119    0.196   10.808    0.000    2.119    0.584
##    .x7               1.960    0.164   11.964    0.000    1.960    0.630
##    .x8               1.986    0.168   11.819    0.000    1.986    0.622
##    .x9               1.556    0.195    7.994    0.000    1.556    0.448
##    .f1               0.836    0.249    3.353    0.001    0.476    0.476
##    .f2               1.574    0.265    5.932    0.000    0.735    0.735
##    .f3               0.627    0.154    4.064    0.000    0.545    0.545
```

```
semPaths(my.fit8, whatLabels="est", color="green")
```

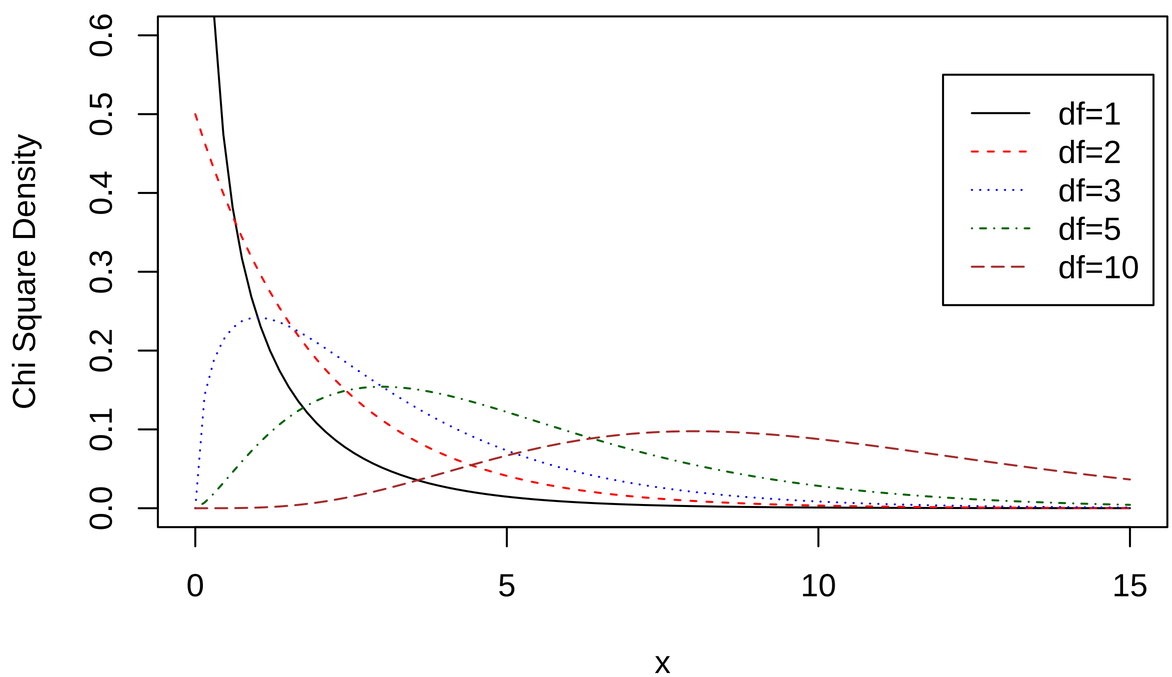


Goodness-of-fit: Chi-square

- Before going into the details, we have to highlight one major difference between SEM and other statistical techniques that we have learned before (Steiger & Fouladi, 1997):
  - Reject-support: Rejecting the null hypothesis supports the researcher's belief.
  - Accept-support: Accepting the null hypothesis supports the researcher's belief.
- Chi-square test (or likelihood ratio) statistic
  - If the proposed model is correct, i.e., $\Sigma = \sum(\theta), (N-1) * F_{min}(S, \hat{\Sigma}(\hat{\theta}))$ follows a chi-square distribution with (p*-q) df.
  - Expected values of chi-squares: $Mean(\chi 2) = df\ and\ Var(\chi 2) = 2df$.

- Actually, it is a badness-of-fit index: large chi-square statistic indicates a poorly fitted model.
- The proposed model is rejected at level of significance if the test statistic is larger than the critical value.

```
curve(dchisq(x,1),xlim=c(0,15),ylim=c(0,0.6),ylab="Chi Square Density")
curve(dchisq(x,2),col="red",lty=2,add=TRUE)
curve(dchisq(x,3),col="blue",lty=3,add=TRUE)
curve(dchisq(x,5),col="dark green",lty=4,add=TRUE)
curve(dchisq(x,10),col="brown",lty=5,add=TRUE)
legend(12,0.55,c("df=1","df=2","df=3","df=5","df=10"),
       col=c("black","red","blue","dark green","brown"),lty=1:5)
```



- There are several "issues" on using a chi-square test statistic to judge the model fit:
  - Model misspecification:
    * Are there any "true" models in the world?
    * The proposed model is correctly specified.
    * In the context of CFA, this means that:
      · There is no double loading;
      · All measurement errors are totally uncorrelated.
  - Violation of the underlying assumptions:
    * Data are normally distributed
    * Large samples are used.

    * When the data are non-normally distributed, especially in clinical studies, or in small sample sizes (e.g., N=100 or 200), the chi-square test statistic may not follow a chi-square distribution.

- Sensitive to the sample size used:

```
-   If there are trivial misspeci cation, all proposed models will be rejected when the sample sizes are
```

```
-   Large samples work against the researcher!
```

- Many SEM users and researchers are aware of the problems of chi-square statistics.
- There are many goodness-of-fit indices proposed as alternative measures.
- There is no single well accepted goodness-of-fit index.
- It is a continuum rather than a discrete choice.

Goodness-of-fit: Absolute fit indices

- They measure the amount of variance and covariance in S that is predicted by the reproduced matrix $\hat{\Sigma}$
- However, they are not rarely used nowadays.
- Goodness of fit index (GFI)
  - It measures the relative amount of variances and covariance in S that are accounted for by the model.
  - Similar to the R^2 in regression analysis.
  - Usually between 0 and 1
- Adjusted goodness of fit index (AGFI)
  - GFI always prefers more complex models.
  - AGFI adjusts for the df in the model.
  - It prefers simpler models with fewer parameters.
  - Similar to the adjusted R2 in regression analysis.

Goodness-of-fit: Incremental fit indices

- They measure the relative improvement in fit by comparing a target model with a baseline model.
- The target model is usually the proposed model, i.e., Σ( ).
- The baseline model is usually a model that all variables are uncorrelated. It is known as the independence model.
- Normed fit index (NFI; Bentler & Bonett, 1980):
  - $NFI = \frac{\chi^2_B - \chi^2_T}{\chi^2_B}$; χ^2_B and χ^2_T are the chi-square statistics of the target and the baseline (or null) models.
  - It measures the proportion reduction in the chi-square values when comparing the baseline to the hypothesized model.

- Non-normed fit index (NNFI; Bentler & Bonett, 1980), also known as Tucker-Lewis index (TLI):
  - $NNFI = \frac{\chi^2_B/df_B - \chi^2_T/df_T}{\chi^2_B/df_B - 1}$; $\mathrm{df_T}$ and $\mathrm{df_B}$ the dfs of the target and the baseline models
  - It adjusts the complexity of the model.
- Comparative fit index (CFI; Bentler, 1990):
  - $CFI = \frac{1 - max[(\chi^2_T - df_T), 0]}{max[(\chi^2_T - df_T), (\chi^2_B - df_B), 0]}$
  - 0<=CFI<=1

What is a well fitted model?

- Rule of thumb (without empirical support): at least > 0.90 (see Lance, Butts, & Michels, 2006 for the historical reasons).
- Goodness-of-fit indices are usually excellent in path models. Why?
- It is hard to provide a single cut-off for all models.

Goodness-of-fit: Residual based indices

- When the proposed model fits the data well, the residuals (difference between model implied covariance matrix and sample covariance matrix) should be small.
- Standardized root mean square residual (SRMR):
  - It measures the average value across the standardized residuals.
  - It ranges from zero (perfect fit) to one (poor fit).
  - Rule of thumb: A well-fitting model < .05.
- Root mean square error of approximation (RMSEA; Steiger & Lind, 1980):
  - Similar to SRMR.
  - Rules of thumb (Browne & Cudeck, 1993):
    * Close fit: < 0.05
    * Reasonable fit: 0.05 – 0.08
    * Inadequate fit: > 0.1
  - Another advantage of using RMSEA is that confidence interval on it is available in many SEM packages.

What do we need to report?

- We usually report the chi-square test statistic, some incremental fit indices and some residual based indices.
- For example, $^2(24) = 103$, $p < .001$; NNFI=0.91, CFI=0.93 and RMSEA=0.11.
- Combinational rules suggested by Hu and Bentler (1999):
  - NNFI (TLI), RNI (not discussed in this class) or CFI > 0.95 and SRMR < .09 or RMSEA < .05 and SRMR < .06
- Although their suggestions are widely accepted, their recommendations are not without challenges (e.g., Marsh, Hau, & Wen, 2004).

Model modification and comparison

- A preferred approach to doing SEM:
  - We may have several theoretically competing models.
  - We are interested in comparing which one is better.
  - If the models are nested, we may use a chi-square difference test to compare them.
  - If the models are non-nested, we may use Akaike information criterion (AIC) or Bayesian information criterion (BIC) to compare them.
- A less preferred approach in doing SEM:
  - After fitting a model, we may want to
    * Simplify the model by deleting some paths;
    * Expand the model by adding more paths.
  - Note. These steps are ad-hoc.

```
summary(my.fit1,fit.measures=TRUE)
```

```
## lavaan 0.6-9 ended normally after 39 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                         9
##
##   Number of observations                           200
##
## Model Test User Model:
##
##   Test statistic                                 1.563
##   Degrees of freedom                                 1
##   P-value (Chi-square)                           0.211
##
## Model Test Baseline Model:
##
##   Test statistic                               132.224
##   Degrees of freedom                                 6
##   P-value                                        0.000
##
## User Model versus Baseline Model:
##
##   Comparative Fit Index (CFI)                    0.996
##   Tucker-Lewis Index (TLI)                       0.973
##
## Loglikelihood and Information Criteria:
##
##   Loglikelihood user model (H0)              -1484.355
##   Loglikelihood unrestricted model (H1)      -1483.574
##
##   Akaike (AIC)                                2986.711
##   Bayesian (BIC)                              3016.396
##   Sample-size adjusted Bayesian (BIC)         2987.883
```

```
##
## Root Mean Square Error of Approximation:
##
##   RMSEA                                          0.053
##   90 Percent confidence interval - lower         0.000
##   90 Percent confidence interval - upper         0.205
##   P-value RMSEA <= 0.05                          0.319
##
## Standardized Root Mean Square Residual:
##
##   SRMR                                           0.016
##
## Parameter Estimates:
##
##   Standard errors                             Standard
##   Information                                 Expected
##   Information saturated (h1) model          Structured
##
## Latent Variables:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   f1 =~
##     x1                1.000
##     x2                1.354    0.378    3.579    0.000
##   f2 =~
##     x3                1.000
##     x4                0.729    0.223    3.274    0.001
##
## Covariances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   f1 ~~
##     f2                0.598    0.199    2.998    0.003
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##    .x1                1.794    0.367    4.888    0.000
##    .x2                0.978    0.595    1.644    0.100
##    .x3                1.051    0.486    2.161    0.031
##    .x4                1.645    0.301    5.465    0.000
##     f1                1.196    0.400    2.992    0.003
##     f2                1.613    0.534    3.019    0.003
```

```
summary(my.fit2,fit.measures=TRUE)
```

```
## lavaan 0.6-9 ended normally after 28 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                         9
##
##   Number of observations                           200
##
## Model Test User Model:
##
##   Test statistic                                 1.563
```

```
##   Degrees of freedom                                 1
##   P-value (Chi-square)                           0.211
##
## Model Test Baseline Model:
##
##   Test statistic                               132.224
##   Degrees of freedom                                 6
##   P-value                                        0.000
##
## User Model versus Baseline Model:
##
##   Comparative Fit Index (CFI)                    0.996
##   Tucker-Lewis Index (TLI)                       0.973
##
## Loglikelihood and Information Criteria:
##
##   Loglikelihood user model (H0)              -1484.355
##   Loglikelihood unrestricted model (H1)      -1483.574
##
##   Akaike (AIC)                                2986.711
##   Bayesian (BIC)                              3016.396
##   Sample-size adjusted Bayesian (BIC)         2987.883
##
## Root Mean Square Error of Approximation:
##
##   RMSEA                                          0.053
##   90 Percent confidence interval - lower         0.000
##   90 Percent confidence interval - upper         0.205
##   P-value RMSEA <= 0.05                          0.319
##
## Standardized Root Mean Square Residual:
##
##   SRMR                                           0.016
##
## Parameter Estimates:
##
##   Standard errors                             Standard
##   Information                                 Expected
##   Information saturated (h1) model          Structured
##
## Latent Variables:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   f1 =~
##     x1                1.094    0.183    5.985    0.000
##     x2                1.480    0.223    6.644    0.000
##   f2 =~
##     x3                1.270    0.210    6.037    0.000
##     x4                0.925    0.170    5.441    0.000
##
## Covariances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   f1 ~~
##     f2                0.431    0.099    4.344    0.000
##
```

```
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##     f1                1.000
##     f2                1.000
##    .x1                1.794    0.367    4.888    0.000
##    .x2                0.978    0.595    1.644    0.100
##    .x3                1.051    0.486    2.161    0.031
##    .x4                1.645    0.301    5.465    0.000
```