Lab Session 4

Course material by Dirk Heerwegh, revised by Ahu Alanya

Spring 2016

Lab Session 4

Lab Session 4: Overview

- Non-normal, continuous data
  - Chi-squared difference testing
- Categorical data
  - Multiple Group Analysis
- Missing data

Non-normal, continuous data

- Robust ML (MLM)
  - Satorra-Bentler scaled χ^2
  - ML parameters estimates with s.e.'s and a mean-adjusted χ^2 test statistic

```
cfa(...,estimator="MLM")
```

- Chi-squared difference testing: use the anova function to account for the fact that the χ^2 value is scaled

Ex. 1 - CFA, non-normal continuous data

- The file "NONML.DAT"contains 5 columns (no header)
- Read in this file and name the variables x1-x5

1.1 Estimate a simple 5 indicator CFA (1 latent variable), without any error covariances. Use ML estimation, then re-estimate using MLM. Compare the output.
1.2 Include an error covariance between x1 and x3 and re-estimate the model. Perform a chi-squared difference test.

Ex. 1 - Solution 1.1.

```
setwd("~/Documents/KUL/SEM/2015")
ds <- read.table("NONML.DAT",header=FALSE,sep=" ",
                 col.names=c("x1","x2","x3","x4","x5"))
model <- "f =~ x1 + x2 + x3 + x4 + x5"
library(lavaan)
fit.ml1 <- cfa(model,data=ds)
fit.mlm1 <- cfa(model,data=ds,estimator="MLM")
```

Ex. 1 - Solution 1.1.

Fit Statistics comparison

Fit Statistic	ML Value	MLM Value
χ^2	87.578	33.092
df	5	5
p	0	0
RMSEA	0.138	0.08
CFI	0.967	0.953
TLI	0.934	0.907

Better fit after accounting for non-normal data. Using the right estimator can also help obtain better fit in Multi-group models.

Ex. 1 - Solution 1.1.

Parameter estimates comparison

Parameter	ML Value (se)	MLM Value (se)
f =~ x1	1 (0)	1 (0)
f =~ x2	0.618 (0.027)	0.618 (0.051)
f =~ x3	1.04 (0.032)	1.04 (0.041)
f =~ x4	0.799 (0.032)	0.799 (0.052)
f =~ x5	0.636 (0.025)	0.636 (0.048)

Note that the loadings remain the same, but standard errors differ.

Ex. 1 - Solution 1.2.

```
model2 <- "f =~ x1 + x2 + x3 + x4 + x5
x1 ~~ x3"
fit.mlm2 <- cfa(model2,data=ds,estimator="MLM")
anova(fit.mlm1,fit.mlm2)
```

```
## Scaled Chi Square Difference Test (method = "satorra.bentler.2001")
##
##          Df   AIC   BIC  Chisq Chisq diff Df diff Pr(>Chisq)
## fit.mlm2  4 15176 15253 25.913
## fit.mlm1  5 15236 15308 87.578     20.587       1  5.698e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Categorical data

- WLSMV (Weighted Least Squares providing a Mean and Variance corrected χ^2 value)

```
cfa(..., ordered=c("y1","y2",...,"y6"), estimator="WLSMV")
# Note that when ordered=... is specified, WLSMV is selected as default
# estimator, so estimator="WLSMV" can be omitted in the call
```

- Will estimate "thresholds" (the underlying y* variables are related to the observed categorical variables by means of these thresholds). This is part of the meanstructure of the model.
- Correlation matrix is used, rather than covariance matrix. No residual variances, the observed categorical variances are estimated.
- Residual variances of y* are estimated (called "scale parameters")

Ex. 2 CFA categorical data

- File BINARY.DAT contains data for 6 variables (y1-y6) which measure alcohol dependence in a sample of 750 participants
- Read in this file (use function read.fwf)
- Fit a simple one-factor CFA model (no error covariances) using WLSMV estimation
- Review the model output (and compare to Brown, p. 393)
- Mplus shows the tetrachoric correlation matrix by default. lavaan does not. Request it using inspect(fit,"sampstat")$cov and compare to the Mplus output shown in Brown, p. 393

Ex. 2 - Solution

```
ds.alc <- read.fwf("BINARY.DAT", widths=rep(1,6),header=FALSE,
                   col.names=paste("y",1:6,sep=""))
model <- "f =~ NA*y1 + y2 + y3 + y4 + y5 + y6
f ~~ 1*f"
fit <- cfa(model,data=ds.alc,ordered=paste("y",1:6,sep=""),estimator="WLSMV")
```

- The WLSMV χ^2 value is printed under "Robust"

Multiple Group CFA

- Measurement invariance
  - Factor loadings and thresholds are fixed or freed in tandem
  - If a factor loading and corresponding threshold(s) are freed, then the scale parameter needs to be fixed to 1
- Syntax

```
y | c(label1,label1)*t1  # set threshold equal across 2 groups
y | c(label1,label2)*t1  # set threshold free across 2 groups
u3 ~*~ c(1,1)*u3 # fix scale of variable "u3" to 1 in both groups
```

Ex. 4

- File "exS4_1" contains responses to 4 categorical variables (y1-y4), each having four response options.
- The variable "group" denotes group membership (G1 and G2)
  4.1. Fit a one-factor model in both groups simultaneously and test configural equivalence
  4.2. Test for measurement non-equivalence (setting factor loadings and thresholds equal across groups)
  4.3. If necessary, relax certain equality constraints and re-fit the model

Ex. 4.1 - Solution

```
ds<-read.table("exS4_1")
model<-'f =~ y1 + y2 + y3 + y4'
fit<-cfa(model,data=ds,group="group",ordered=c("y1","y2","y3","y4"))

fit2<-cfa(model,data=ds,group="group",ordered=c("y1","y2","y3","y4"),
          group.equal=c("loadings","thresholds"))
#summary(fit2,mod=T)
```

Ex. 4.2 - Solution

```
model2 <- 'f =~ y1 + y2 + y3 + y4 y3
~*~ c(1,1)*y3'
fit3<-cfa(model2,data=ds,group="group",ordered=c("y1","y2","y3","y4"),
          group.equal=c("loadings","thresholds"),
          group.partial=c("f =~ y3","y3 | t1","y3 | t2","y3 | t3"))
#summary(fit3,mod=T)
```

Ex. 5

This example is from Hirschfeld, and von Brachel's tutorial1 Data is from an online survey on sexual compulsivity scale which is available on http://personality-testing.info/_rawdata/. The scale consists of ten items regarding descriptions about sexual behaviour, e.g "I think about sex more than I would like to". The items are measured on a four-category likert scale ranging from "not at all like me" to "very much like me".

- Subset the data so that you have only codes 1 and 2 for gender variables for the analysis
  5.1. Fit a one-factor model of sexual compulsivity in both groups simultaneously and test configural equivalence
  5.2. Test for measurement non-equivalence (setting factor loadings and thresholds equal across groups)
  5.3. If necessary, relax certain equality constraints and re-fit the model.

Ex. 5.1 – Solution

```
##Categorical indicators
library(lavaan)
library(semTools)
library(semPlot)
#download.file("http://personality-testing.info/_rawdata/SCS.zip","SCS.zip")
unzip("SCS.zip")
scs <- read.csv("SCS/data.csv")
scs <- subset(scs, gender == "1" | gender == "2")
scs_model <- 'scs =~ Q1 + Q2 + Q3 + Q4 + Q5 + Q6 + Q7 + Q8 + Q9 + Q10'

# Both groups
scs_model_fit <- cfa(scs_model, ordered = c("Q1", "Q2", "Q3", "Q4", "Q5", "Q6", "Q7", "Q8",
"Q9", "Q10"), data=scs)
summary(scs_model_fit, fit.measures = TRUE)

semPaths(scs_model_fit, "std", rotation = 2, layout = "tree2", nCharNodes =
           0, sizeLat = 15, sizeLat2 = 7, label.norm = "OOOOO", mar=c(2,-4,2,4),
         curvePivot = TRUE, edge.label.cex=1.2, residuals = FALSE, thresholds = FALSE)
```

Ex. 5.2 – Solution

```
scs_model_config <- cfa(scs_model, ordered = c("Q1", "Q2", "Q3", "Q4", "Q5", "Q6", "Q7", "Q8",
"Q9", "Q10"), group = "gender", data=scs)

config <- cfa(scs_model, ordered = c("Q1", "Q2", "Q3", "Q4", "Q5","Q6", "Q7", "Q8", "Q9",
"Q10"), group = "gender", group.equal =c("loadings"), data=scs)

metric <- cfa(scs_model, ordered = c("Q1", "Q2", "Q3", "Q4", "Q5", "Q6", "Q7", "Q8", "Q9",
"Q10"), group = "gender", group.equal = c("loadings", "thresholds"), data=scs)
```

[1] *Practical Assessment, Research & Evaluation, Vol 19, No 7*

```
# Compare models:
anova(config, metric)
measurementInvariance(scs_model, data=scs, group="gender")
#or# lavTestLRT(config, metric)
# And look at the CFI difference whether it is >0.01
fitMeasures(config, "cfi") - fitMeasures(metric, "cfi")
#or
fitMeasures(config, c("cfi","cfi.scaled")) - fitMeasures(metric, c("cfi","cfi.scaled"))

#Alternative way for model comparison
measurementInvariance(scs_model, data=scs, group="gender", strict=TRUE)
```

Missing data

- Default = listwise deletion
- Best option: Direct ML (Full Information ML)
- Second best option: Multiple imputation
- See Brown, Chapter 9
- Libraries, Syntax:

```
# lavaan supports direct ML (FIML)
cfa(model,data=...,missing="direct")

# MI: use "amelia" function from the Amelia library to generate m imputed datasets
out<-amelia(input.dataset,m=5)  # m=5: 5 imputed datasets will be stored in "out"
# analyze with "runMI" from the semTools library (which in turn relies on lavaan)
out.mi<-runMI(model,out$imputations,chi="all",fun="cfa",estimator="ML")
```

Terms

Values in a data set are **missing completely at random** (MCAR) if the events that lead to any particular data-item being missing are independent both of observable variables and of unobservable parameters of interest, and occur entirely at random. When data are MCAR, the analyses performed on the data are unbiased; however, data are rarely MCAR.

Missing at random
Missing at random (MAR) occurs when the missingness is not random, but where missingness can be fully accounted for by variables on which there is complete information. MAR is an assumption that is impossible to verify statistically, we must rely on its substantive reasonableness.

Missing not at random
Missing not at random (MNAR) (also known as nonignorable nonresponse) is data that is neither MAR nor MCAR (i.e. the value of the variable that's missing is related to the reason it's missing).

Missing data - Ex

- The file “cfamiss.dat” contains 5 variables (a subject identifier, s1-s4). The four “s”-variables measure 1 latent trait (“esteem”). For substantive reasons, s2 and s4 have correlated error variances.
- Fit the model using FIML
- Fit the model using MI (generate 5 imputed datasets)

Missing data - Direct ML (FIML)

```
ds<-read.table("cfamiss.dat",col.names=c("id","s1","s2","s3","s4"),na=9)
model <- "esteem =~ s1 + s2 + s3 + s4 s2
~~ s4"
fit <- cfa(model,data=ds,missing="direct")
summary(fit)
```

Results with ML

Note that the default in Lavaan is to use listwise deletion therefore 385 observation are used in the model.

```
 Used       Total
  Number of observations                          385         650

Estimator                                       ML
  Minimum Function Test Statistic              0.108
  Degrees of freedom                               1
  P-value (Chi-square)                         0.743

Parameter Estimates:

  Information                                 Expected
  Standard Errors                             Standard

Latent Variables:
                   Estimate  Std.Err  Z-value  P(>|z|)   Std.lv  Std.all
  esteem =~
    s1                1.000                               0.919    0.737
    s2                1.411    0.081   17.410    0.000    1.296    0.931
    s3                1.361    0.074   18.495    0.000    1.251    0.880
    s4                1.273    0.075   16.963    0.000    1.170    0.910
```

Results with FIML fit2

```
Number of observations                          650

  Number of missing patterns                      5

  Estimator                                       ML
  Minimum Function Test Statistic              1.266
  Degrees of freedom                               1
  P-value (Chi-square)                         0.260

Parameter Estimates:

  Information                                 Observed
  Standard Errors                             Standard

Latent Variables:
                   Estimate  Std.Err  Z-value  P(>|z|)   Std.lv  Std.all
  esteem =~
    s1                1.000                               0.909    0.737
    s2                1.383    0.064   21.714    0.000    1.257    0.920
    s3                1.360    0.059   23.030    0.000    1.235    0.880
    s4                1.275    0.063   20.092    0.000    1.158    0.905
```

FIML has lower standard errors. Why? Because FIML makes more efficient use of the data at hand and preserves statistical power (lower standard errors). Therefore, even if your missing pattern is MCAR, it is a better option than listwise deletion.

Missing data - Multiple Imputation

```
library(Amelia)
# create 5 imputed datasets
a.out<-amelia(subset(ds,select=c("s1","s2","s3","s4")),m=5)
```

```
## -- Imputation 1 --
##
##   1  2  3  4  5
##
## -- Imputation 2 --
##
##   1  2  3  4  5  6  7
##
## -- Imputation 3 --
##
##   1  2  3  4  5  6  7
##
## -- Imputation 4 --
##
##   1  2  3  4  5  6
##
## -- Imputation 5 --
##
##   1  2  3  4  5  6
```

Missing data - Multiple Imputation

```
# analyze using lavaan via semTools
library(semTools)
out.mi<-runMI(model,a.out$imputations,chi="all",fun="cfa",estimator="ML")
summary(out.mi)
```

```
lavaan (0.5-20) converged normally after   5 iterations

  Number of observations                           650

  Estimator                                         ML
  Minimum Function Test Statistic                1.254
  Degrees of freedom                                 1
  P-value (Chi-square)                           0.263

Parameter Estimates:

  Information                                 Expected
  Standard Errors                             Standard

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

```
  esteem =~
    s1                1.000
    s2                1.373    0.063   21.700    0.000
    s3                1.345    0.060   22.455    0.000
    s4                1.275    0.071   18.082    0.000

Covariances:
                   Estimate  Std.Err  Z-value  P(>|z|)
  s2 ~~
    s4               -0.256    0.041   -6.293    0.000

Variances:
                   Estimate  Std.Err  Z-value  P(>|z|)
    s1                0.689    0.043   15.968    0.000
    s2                0.284    0.043    6.637    0.000
    s3                0.444    0.041   10.880    0.000
    s4                0.281    0.052    5.414    0.000
    esteem            0.838    0.080   10.534    0.000
```

```
# Get missing data patterns and covariance coverage similar
# to that found in Mplus output.
inspect(fit, 'patterns')
inspect(fit, 'coverage')
```

Please inform yourself more about the Amelia package before you perfom Multiple Imputation. For example.,

Binary variables

For binary variables you can specify the nominals option in Amelia function

Ordinal variables

“Users are advised to allow Amelia to impute non-integer values for any missing data, and to use these non-integer values in their analysis. Sometimes this makes sense, and sometimes this defies intuition.”

Variables to include in the imputation model

“When performing multiple imputation, the first step is to identify the variables to include in the imputation model. It is crucial to include at least as much information as will be used in the analysis model. That is, any variable that will be in the analysis model should also be in the imputation model. This includes any transformations or interactions of variables that will appear in the analysis model.”