## Measurement invariance excercise

#In using R, we first need to load packages we want to use for calculations. We will use *lavaan* and #we need *haven* to load our data

library (lavaan)

library (haven)

#with this line of code we are loading the data and saving it to the R object we called data

#be careful that your data is in the same folder as your R syntax

data<- read\_spss ("PROBIOPS dataset for workshop 2023 only.sav")

#Measurement invariance of self-esteem-----------

#configural invariance

#with this chunk of code we are specifying our model of self esteem configural invariance (**the same #syntax we used in JASP**, but now it is in quotation marks) and saving it to an object called se\_conf

se\_conf<-

'# Factors

SE\_t1 =~ seload\_1\_1\*a24x1 + seload\_1\_2\*a24x2 + seload\_1\_3\*a24x3 + seload\_1\_4\*a24x4

SE\_t3 =~ seload\_2\_1\*c24x1 + seload\_2\_2\*c24x2 + seload\_2\_3\*c24x3 + seload\_2\_4\*c24x4

# Residual Correlations

a24x1 ~~ c24x1

a24x2 ~~ c24x2

a24x3 ~~ c24x3

a24x4 ~~ c24x4'

#with the following line of code we are estimating our model and saving it to an R object called fit.se\_conf

#we use the cfa () function with model, data and meanstructure arguments (the function has other #arguments as well which can be changed. If we do not change them, the cfa function will use their #default values

#in model argument we enter the R object se\_conf in which our model specification is saved

#in data argument we enter the R object data in which our dataset is saved

#by saying that meanstructure argument is TRUE we are saying lavaan that we want it to estimate #intercepts of items

fit.se\_conf<- cfa (model = se\_conf, data = data, meanstructure =TRUE)

#with the summary () function we are telling R to give us the output of our analysis

#in object argument we enter the R object fit.se\_conf in which we saved the results of our analysis

#by saying that standardized argument is TRUE we are saying to R that we want it to output the #standardized estimates of parameters

#by saying that fit.measures argument is TRUE (we can also use just T instead of TRUE) we are telling #R to include all global fit estimates in the output

summary (object = fit.se\_conf, standardized = TRUE, fit.measures = T)

#metric invariance

#model specification (**the same syntax we used in JASP)**

se\_metric<-

'# Factors

SE\_t1 =~ seload\_1\_1\*a24x1 + seload\_1\_2\*a24x2 + seload\_1\_3\*a24x3 + seload\_1\_4\*a24x4

SE\_t3 =~ seload\_1\_1\*c24x1 + seload\_1\_2\*c24x2 + seload\_1\_3\*c24x3 + seload\_1\_4\*c24x4

# Residual Correlations

a24x1 ~~ c24x1

a24x2 ~~ c24x2

a24x3 ~~ c24x3

a24x4 ~~ c24x4'

#model calculation

fit.se\_metric<- cfa (se\_metric, data, meanstructure =T)

#model output

summary (fit.se\_metric, standardized = T, fit.measures = T)

#scalar invariance

#model specification (**the same syntax we used in JASP)**

se\_scalar<-

'# Factors

SE\_t1 =~ seload\_1\_1\*a24x1 + seload\_1\_2\*a24x2 + seload\_1\_3\*a24x3 + seload\_1\_4\*a24x4

SE\_t3 =~ seload\_1\_1\*c24x1 + seload\_1\_2\*c24x2 + seload\_1\_3\*c24x3 + seload\_1\_4\*c24x4

# Residual Correlations

a24x1 ~~ c24x1

a24x2 ~~ c24x2

a24x3 ~~ c24x3

a24x4 ~~ c24x4

#intercepts

a24x1~ i1\*1

a24x2~ i2\*1

a24x3~ i3\*1

a24x4~ i4\*1

c24x1~ i1\*1

c24x2~ i2\*1

c24x3~ i3\*1

c24x4~ i4\*1

#SE\_t1~ 0\*1

#SE\_t3~ 1

'

#model calculation

fit.se\_scalar<- cfa (se\_scalar, data, meanstructure =T)

#model output

summary (fit.se\_scalar, standardized = T, fit.measures = T)

#residual invariance

#model specification

se\_resid<-

'# Factors

SE\_t1 =~ seload\_1\_1\*a24x1 + seload\_1\_2\*a24x2 + seload\_1\_3\*a24x3 + seload\_1\_4\*a24x4

SE\_t3 =~ seload\_1\_1\*c24x1 + seload\_1\_2\*c24x2 + seload\_1\_3\*c24x3 + seload\_1\_4\*c24x4

# Residual Correlations

a24x1 ~~ c24x1

a24x2 ~~ c24x2

a24x3 ~~ c24x3

a24x4 ~~ c24x4

#intercepts

a24x1~ i1\*1

a24x2~ i2\*1

a24x3~ i3\*1

a24x4~ i4\*1

c24x1~ i1\*1

c24x2~ i2\*1

c24x3~ i3\*1

c24x4~ i4\*1

#residuals equalized

a24x1~~ r1\*a24x1

a24x2~~ r2\*a24x2

a24x3~~ r3\*a24x3

a24x4~~ r4\*a24x4

c24x1~~ r1\*c24x1

c24x2~~ r2\*c24x2

c24x3~~ r3\*c24x3

c24x4~~ r4\*c24x4

'

#model calculation

fit.se\_resid<- cfa (se\_resid, data, meanstructure =T)

#model output

summary (fit.se\_resid, standardized = T, fit.measures = T)

#with the anova () function we can compare the fit of different models with chi-square difference #test

#arguments in the anova () function are R objects in which we saved the model calculations

anova (fit.se\_conf, fit.se\_metric, fit.se\_scalar, fit.se\_resid)

#measurement invariance of depression and anxiety --------

#configural invariance

#model specification

DA\_conf<-

'# Factors

DA\_t1 =~ daload\_1\_1\*a22x1 + daload\_1\_2\*a22x2 + daload\_1\_3\*a22x3 + daload\_1\_4\*a22x4

DA\_t3 =~ daload\_2\_1\*c22x1 + daload\_2\_2\*c22x2 + daload\_2\_3\*c22x3 + daload\_2\_4\*c22x4

# Residual Correlations

a22x1 ~~ c22x1

a22x2 ~~ c22x2

a22x3 ~~ c22x3

a22x4 ~~ c22x4'

#model calculation

fit.DA\_conf<- cfa (DA\_conf, data, meanstructure =T)

#model output

summary (fit.DA\_conf, standardized = T, fit.measures = T)

#metric invariance

#model specification

DA\_metric<-

'# Factors

DA\_t1 =~ daload\_1\_1\*a22x1 + daload\_1\_2\*a22x2 + daload\_1\_3\*a22x3 + daload\_1\_4\*a22x4

DA\_t3 =~ daload\_1\_1\*c22x1 + daload\_1\_2\*c22x2 + daload\_1\_3\*c22x3 + daload\_1\_4\*c22x4

# Residual Correlations

a22x1 ~~ c22x1

a22x2 ~~ c22x2

a22x3 ~~ c22x3

a22x4 ~~ c22x4'

#model calculation

fit.DA\_metric<- cfa (DA\_metric, data, meanstructure =T)

#model output

summary (fit.DA\_metric, standardized = T, fit.measures = T)

#scalar invariance

#model specification

DA\_scalar<-

'# Factors

DA\_t1 =~ daload\_1\_1\*a22x1 + daload\_1\_2\*a22x2 + daload\_1\_3\*a22x3 + daload\_1\_4\*a22x4

DA\_t3 =~ daload\_1\_1\*c22x1 + daload\_1\_2\*c22x2 + daload\_1\_3\*c22x3 + daload\_1\_4\*c22x4

# Residual Correlations

a22x1 ~~ c22x1

a22x2 ~~ c22x2

a22x3 ~~ c22x3

a22x4 ~~ c22x4

#intercepts

a22x1~ i1\*1

a22x2~ i2\*1

a22x3~ i3\*1

a22x4~ i4\*1

c22x1~ i1\*1

c22x2~ i2\*1

c22x3~ i3\*1

c22x4~ i4\*1

#DA\_t1~ 0\*1

#DA\_t3~ 1

'

#model calculation

fit.DA\_scalar<- cfa (DA\_scalar, data, meanstructure =T)

#model output

summary (fit.DA\_scalar, standardized = T, fit.measures = T)

#model comparisons

anova (fit.DA\_conf, fit.DA\_metric, fit.DA\_scalar)

## Cross-lagged model with two variables and two time-points

#loading the necessary packages

library (lavaan)

library (haven)

#loading data in an R object named data

data<- read\_spss ("PROBIOPS dataset for workshop 2023 only.sav")

#specifying the model (**the same syntax we used in JASP)**

cl\_model<-

'

#Self-esteem measurement model

# Factors

SE\_t1 =~ seload\_1\_1\*a24x1 + seload\_1\_2\*a24x2 + seload\_1\_3\*a24x3 + seload\_1\_4\*a24x4

SE\_t3 =~ seload\_1\_1\*c24x1 + seload\_1\_2\*c24x2 + seload\_1\_3\*c24x3 + seload\_1\_4\*c24x4

# Residual Correlations

a24x1 ~~ c24x1

a24x2 ~~ c24x2

a24x3 ~~ c24x3

a24x4 ~~ c24x4

#Depression and anxiety measurement model

# Factors

DA\_t1 =~ daload\_1\_1\*a22x1 + daload\_1\_2\*a22x2 + daload\_1\_3\*a22x3 + daload\_1\_4\*a22x4

DA\_t3 =~ daload\_1\_1\*c22x1 + daload\_1\_2\*c22x2 + daload\_1\_3\*c22x3 + daload\_1\_4\*c22x4

# Residual Correlations

a22x1 ~~ c22x1

a22x2 ~~ c22x2

a22x3 ~~ c22x3

a22x4 ~~ c22x4

#cross-lagged regression

DA\_t3~ clda\*SE\_t1+ autoregda\*DA\_t1

SE\_t3~ clse\*DA\_t1+ autoregse\*SE\_t1'

#model calculcation (notice that we are not specifying the meanstructure argument to be TRUE #anymore

fit.cl\_model<- sem (cl\_model, data)

#model output

summary (fit.cl\_model, fit.measures =T, standardized = T)

## Mediation (with syntax for half-longitudinal mediation)

#loading the necessary packages

library (lavaan)

library (haven)

#loading the data into R object named data

data<- read\_spss ("PROBIOPS dataset for workshop 2023 only.sav")

#half-longitudinal mediation--------

#specifying the model – for a diagram of this model check out the half-longitudinal slide in the cross-#lagged presentation

med\_hlong<- '

#predictor-> mediator

c\_selfesteem~ a\*a82 + autoregse\*a\_selfesteem + clrevse\*a\_depanx

#mediator-> outcome (+ direct effect)

c\_depanx~ b\*a\_selfesteem + c\*a82 + autoregda\*a\_depanx

#reverse direction soc media use

c82~ clrevsmu\*a\_selfesteem + autoregsmu\*a82

#calculating indirect and total effects

ind := a\*b

total := ind + c

'

#model calculation. Notice that we added two arguments

#in se = “boot” argument we are saying to lavaan to calculate boostrapped standard errors

#in bootstrap argument we are saying to lavaan that we want it to estimate 1000 bootstrap samples

fit.med\_hlong <- sem (med\_hlong, data, se = "boot", bootstrap = 1000)

#model output

summary (fit.med\_hlong, standardized = T, fit.measures = T)

#as model output will not give the confidence intervals of effects, we need to view use the #parameterestimates () function with an argument boot.ci.type = “bca.simple”

parameterestimates(object = fit.med\_hlong, boot.ci.type = "bca.simple")

#full-longitudinal mediation--------

#model specification (**the same syntax we used in JASP)**

med\_full<-

##t2 prediction

#predictor-> mediator

b\_selfesteem~ a\*a82 + autoregse1\*a\_selfesteem + clrevse1\*a\_depanx

#reverse direction soc media use

b82~ clrevsmu1\*a\_selfesteem+ autoregsmu1\*a82

#autoregression dep-anx

b\_depanx~ autoregda1\*a\_depanx+ a\_selfesteem

##t3 prediction

#mediator-> outcome (+ direct effect)

c\_depanx~ b\*b\_selfesteem + c\*a82 + autoregda2\*b\_depanx

#reverse direction social media use

c82~ clrevsmu2\*b\_selfesteem + autoregsmu2\*b82

#reverse direction self-esteem

c\_selfesteem~ clrevse2\*b\_depanx+ autoregse2\*b\_selfesteem+ b82

#calculating indirect and total effects

indirect := a\*b

total := indirect + c '

#model calculation. Notice that we added two arguments

#in se = “boot” argument we are saying to lavaan to calculate boostrapped standard errors

#in bootstrap argument we are saying to lavaan that we want it to estimate 1000 bootstrap samples

fit.med\_full <- sem (med\_full, data, se = "boot", bootstrap = 1000)

#model output

summary (fit.med\_full, standardized = T, fit.measures = T)

#as model output will not give the confidence intervals of effects, we need to view use the #parameterestimates () function with an argument boot.ci.type = “bca.simple”

parameterestimates(fit.med\_full, boot.ci.type = "bca.simple")