Structural Equation Modeling

P.03 - Model Parameters

November 01, 2022

Lab Description

For this practical you will need the following packages: `lavaan` and `semPlot`. You can install and load these packages using the following code:

```
# Install packages.
install.packages(c("lavaan", "semPlot"))

# Load the packages.
library(lavaan)
library(semPlot)
```

Exercise 1

Umstattd Meyer et al. (2014) measured poor psychosocial health as a single factor model using three item facets from a depression questionnaire and a measure of social activity. The covariance matrix is given in Figure 1.

	D1	D2	D3	SA
Depression 1	0.77	0.38	0.39	−0.25
Depression 2	0.38	0.65	0.39	−0.32
Depression 3	0.39	0.39	0.62	−0.27
Social Activity	−0.25	−0.32	−0.27	6.09

Data taken from Umstattd-Meyer et al. (2013, pp. 4-5)

Figure 1: Covariances for exercise 1 ($N = 6053$).

a. Enter the covariance matrix into `R`.

```
# Input the covariance matrix.
covariances <- c(0.77, 0.38, 0.65, 0.39, 0.39, 0.62, -0.25, -0.32, -0.27, 6.09)

# Create the covariance matrix using "lavaan::lav_matrix_lower2full".
covariances <- lav_matrix_lower2full(covariances)
```

```r
# Add row and column names for the variables.
# Dep1 stands for "Depression 1".
# Dep2 stands for "Depression 2".
# Dep3 stands for "Depression 3".
# SocAct stands for "Social Activity".
rownames(covariances) <- colnames(covariances) <- c("Dep1", "Dep2", "Dep3", "SocAct")
```

b. Fit the model using (1) the marker variable approach, (2) the standardized latent variable approach, and (3) the effect coding approach for achieving identification of the latent variable. For the marker variable method, use `Depression 1` as the marker variable. The resulting χ^2 and degrees of freedom (DF) should be identical for the three models.

Using (1) the marker variable approach.

```r
# Model syntax.
# `PsychoSocial` (i.e., psychosocial health) in the name we selected for the latent construct.
model_marker <- "
    PsychoSocial =~ Dep1 + Dep2 + Dep3 + SocAct
"

# Model fit.
model_marker_fit <- cfa(model_marker, sample.cov = covariances, sample.nobs = 6053)

# Model summary.
summary(model_marker_fit, standardized = TRUE)
```

```
## lavaan 0.6-12 ended normally after 27 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                         8
##
##   Number of observations                          6053
##
## Model Test User Model:
##
##   Test statistic                                 9.620
##   Degrees of freedom                                 2
##   P-value (Chi-square)                           0.008
##
## 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
##   PsychoSocial =~
##     Dep1              1.000                               0.616    0.701
##     Dep2              1.005    0.021   47.588    0.000    0.619    0.768
```

```
##     Dep3              1.025    0.022   47.638    0.000    0.631    0.801
##     SocAct           -0.736    0.058  -12.793    0.000   -0.453   -0.184
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##    .Dep1              0.391    0.009   41.276    0.000    0.391    0.508
##    .Dep2              0.267    0.008   33.581    0.000    0.267    0.411
##    .Dep3              0.222    0.008   28.886    0.000    0.222    0.358
##    .SocAct            5.884    0.108   54.559    0.000    5.884    0.966
##     PsychoSocial      0.379    0.014   27.888    0.000    1.000    1.000
```

Note. By default, the first indicator of each latent variable is used as the marker variable, i.e., `Depression 1` in our case. However, we may select which indicator to use as the marker variable in the `lavaan` syntax.

Using (2) the standardized latent variable approach.

```
# Model syntax.
model_stdlv <- "
    PsychoSocial =~ NA * Dep1 + Dep1 + Dep2 + Dep3 + SocAct
    PsychoSocial ~~ 1 * PsychoSocial
"

# Model fit.
model_stdlv_fit_1 <- cfa(model_stdlv, sample.cov = covariances, sample.nobs = 6053)

# Model summary.
summary(model_stdlv_fit_1, standardized = TRUE)
```

```
## lavaan 0.6-12 ended normally after 19 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                         8
##
##   Number of observations                          6053
##
## Model Test User Model:
##
##   Test statistic                                 9.620
##   Degrees of freedom                                 2
##   P-value (Chi-square)                           0.008
##
## 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
##   PsychoSocial =~
##     Dep1              0.616    0.011   55.776    0.000    0.616    0.701
```

```
##     Dep2              0.619    0.010   61.392    0.000    0.619    0.768
##     Dep3              0.631    0.010   64.285    0.000    0.631    0.801
##     SocAct           -0.453    0.035  -12.967    0.000   -0.453   -0.184
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##     PsychoSocial      1.000                               1.000    1.000
##    .Dep1              0.391    0.009   41.276    0.000    0.391    0.508
##    .Dep2              0.267    0.008   33.581    0.000    0.267    0.411
##    .Dep3              0.222    0.008   28.886    0.000    0.222    0.358
##    .SocAct            5.884    0.108   54.559    0.000    5.884    0.966
```

Instead of tweaking the model syntax, we can also indicate that we want to standardize the latent variable by setting the `std.lv = TRUE` argument in `lavaan::cfa`. In this case, we use the model syntax `model_marker`.

```
# Model fit using standardized latent variable approach via `std.lv = TRUE`.
model_stdlv_fit_2 <- cfa(
    model_marker,
    sample.cov = covariances,
    std.lv = TRUE,
    sample.nobs = 6053
)

# Model summary.
summary(model_stdlv_fit_2, standardized = TRUE)
```

```
## lavaan 0.6-12 ended normally after 19 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                         8
##
##   Number of observations                          6053
##
## Model Test User Model:
##
##   Test statistic                                 9.620
##   Degrees of freedom                                 2
##   P-value (Chi-square)                           0.008
##
## 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
##   PsychoSocial =~
##     Dep1              0.616    0.011   55.776    0.000    0.616    0.701
##     Dep2              0.619    0.010   61.392    0.000    0.619    0.768
##     Dep3              0.631    0.010   64.285    0.000    0.631    0.801
```

```
##     SocAct           -0.453    0.035  -12.967    0.000   -0.453   -0.184
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##    .Dep1              0.391    0.009   41.276    0.000    0.391    0.508
##    .Dep2              0.267    0.008   33.581    0.000    0.267    0.411
##    .Dep3              0.222    0.008   28.886    0.000    0.222    0.358
##    .SocAct            5.884    0.108   54.559    0.000    5.884    0.966
##     PsychoSocial      1.000                               1.000    1.000
```

Both ways of standardizing the latent variable show in the output that the variance of the latent variable `PsychoSocial` has a variance of 1. Furthermore, model fit is identical to marker variable approach (i.e., $\chi^2 = 9.620$).

Using (3) the effect coding approach.

```r
# Model syntax.
model_effect_coding <- "
    PsychoSocial =~
        NA * Dep1 +
        LoadingDep1 * Dep1 +
        LoadingDep2 * Dep2 +
        LoadingDep3 * Dep3 +
        LoadingSocAct * SocAct

    # Effect coding.
    LoadingDep1 == 4 - LoadingDep2 - LoadingDep3 - LoadingSocAct
"

# Model fit.
model_effect_coding_fit <- cfa(model_effect_coding, sample.cov = covariances, sample.nobs = 6053)

# Summary.
summary(model_effect_coding_fit, standardized = TRUE)
```

```
## lavaan 0.6-12 ended normally after 29 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                         9
##   Number of equality constraints                     1
##
##   Number of observations                          6053
##
## Model Test User Model:
##
##   Test statistic                                 9.620
##   Degrees of freedom                                 2
##   P-value (Chi-square)                           0.008
##
## 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
##   PsychoSocial =~
##     Dep1     (LdD1)    1.744    0.048   36.449    0.000    0.616    0.701
##     Dep2     (LdD2)    1.753    0.048   36.367    0.000    0.619    0.768
##     Dep3     (LdD3)    1.787    0.049   36.377    0.000    0.631    0.801
##     SocAct   (LdSA)   -1.284    0.131   -9.832    0.000   -0.453   -0.184
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##    .Dep1              0.391    0.009   41.276    0.000    0.391    0.508
##    .Dep2              0.267    0.008   33.581    0.000    0.267    0.411
##    .Dep3              0.222    0.008   28.886    0.000    0.222    0.358
##    .SocAct            5.884    0.108   54.559    0.000    5.884    0.966
##     PsychoSocial      0.125    0.007   17.914    0.000    1.000    1.000
##
## Constraints:
##                                                |Slack|
##     LoadingDep1-(4-LdngDp2-LdngDp3-LdngScAct)    0.000
```

```r
# Extract the values for the loadings.
loadings <- coef(model_effect_coding_fit)[1:4]

# Compute the mean for the loadings.
mean(loadings)
```

```
## [1] 1
```

Again, we see that model fit is identical to the marker and standardized latent variables approaches. Latent variable is on the same scale as the average of all the indicators (i.e., optimally weighted average of set of indicators). Average of the loadings equals $\frac{1.744+1.753+1.787-1.284}{4} = 1$. The variance of `PsychoSocial` latent variable is 0.125, and it represents the average of the amount of reliable variance that each indicator contributes to the definition of this latent construct.

Similar to the *standardized latent variable* approach, we can use the effects coding approach by specifying the `effect.coding = TRUE` argument in `lavaan` instead of tweaking the model syntax as above. In this case, we use the model syntax stored in variable `model_marker`. The `lavaan` documentation for the argument `effect.coding` (i.e., see `?lavOptions`) gives us more information on how the effects coding is achieved, i.e.:

> Can be logical or character string. If logical and TRUE, this implies `effect.coding = c("loadings", "intercepts")`. If logical and `FALSE`, it is set equal to the empty string. If `"loadings"` is included, equality constraints are used so that the average of the factor loadings (per latent variable) equals 1. Note that this should not be used together with `std.lv = TRUE`. If `"intercepts"` is included, equality constraints are used so that the sum of the intercepts (belonging to the indicators of a single latent variable) equals zero. As a result, the latent mean

will be freely estimated and usually equal the average of the means of the involved indicators.

```
# Model fit using effects coding via `effect.coding = TRUE`.
model_effect_coding_fit_2 <- cfa(
    model_marker,
    sample.cov = covariances,
    effect.coding = TRUE,
    sample.nobs = 6053
)
```

```
## Warning in lav_partable_add_bounds(partable = lavpartable, lavpta = lavpta, :
## lavaan WARNING: automatic bounds not available (yet) if effect.coding is used

## Warning in lav_partable_add_bounds(partable = lavpartable, lavh1 = lavh1, :
## lavaan WARNING: automatic bounds not available (yet) if effect.coding is used
```

```
# Model summary.
summary(model_effect_coding_fit_2, standardized = TRUE)
```

```
## lavaan 0.6-12 ended normally after 29 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                         9
##   Number of equality constraints                     1
##
##   Number of observations                          6053
##
## Model Test User Model:
##
##   Test statistic                                 9.620
##   Degrees of freedom                                 2
##   P-value (Chi-square)                           0.008
##
## 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
##   PsychoSocial =~
##     Dep1              1.744    0.048   36.449    0.000    0.616    0.701
##     Dep2              1.753    0.048   36.367    0.000    0.619    0.768
##     Dep3              1.787    0.049   36.377    0.000    0.631    0.801
##     SocAct           -1.284    0.131   -9.832    0.000   -0.453   -0.184
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##    .Dep1              0.391    0.009   41.276    0.000    0.391    0.508
##    .Dep2              0.267    0.008   33.581    0.000    0.267    0.411
##    .Dep3              0.222    0.008   28.886    0.000    0.222    0.358
##    .SocAct            5.884    0.108   54.559    0.000    5.884    0.966
```

```
##     PsychoSocial      0.125    0.007   17.914    0.000    1.000    1.000
```

c. Re-estimate the the first model (i.e., using the marker variable method), but now with the additional equality constraints between the loadings of `Depression 1`, `Depression 2`, and `Social Activity`.

```
# Model syntax with equality constraint.
model_marker_constrained <- "
    PsychSocLV =~ c1 * Dep1 + c1 * Dep2 + Dep3 + c1 * SocAct
"

# Model fit.
model_marker_constrained_fit <- cfa(model_marker_constrained, sample.cov = covariances, sample.nobs = 6053)

# Model summary.
summary(model_marker_constrained_fit)
```

```
## lavaan 0.6-12 ended normally after 19 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                         6
##
##   Number of observations                          6053
##
## Model Test User Model:
##
##   Test statistic                               866.417
##   Degrees of freedom                                 4
##   P-value (Chi-square)                           0.000
##
## Parameter Estimates:
##
##   Standard errors                             Standard
##   Information                                 Expected
##   Information saturated (h1) model          Structured
##
## Latent Variables:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   PsychSocLV =~
##     Dep1      (c1)    1.000
##     Dep2      (c1)    1.000
##     Dep3              1.102    0.021   52.812    0.000
##     SocAct    (c1)    1.000
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##    .Dep1              0.403    0.009   43.756    0.000
##    .Dep2              0.288    0.008   37.834    0.000
##    .Dep3              0.208    0.008   25.245    0.000
##    .SocAct            6.732    0.124   54.458    0.000
##     PsychSocLV        0.339    0.010   35.255    0.000
```

d. Test the constrained against the unconstrained marker model using the likelihood ratio test. What do you conclude?

```r
# Perform a likelihood ratio test using the `anova` function in `R`.
anova(model_marker_constrained_fit, model_marker_fit)
```

```
## Chi-Squared Difference Test
##
##                              Df   AIC   BIC    Chisq Chisq diff Df diff
## model_marker_fit              2 66677 66731   9.6199
## model_marker_constrained_fit  4 67530 67570 866.4168      856.8       2
##                              Pr(>Chisq)
## model_marker_fit
## model_marker_constrained_fit  < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We see that the constrained model fits worse than unconstrained model, hence we prefer the unconstrained model.

Exercise 2

Consider the following hypothesized four-factor CFA model of self-concept depicted in Figure 2.

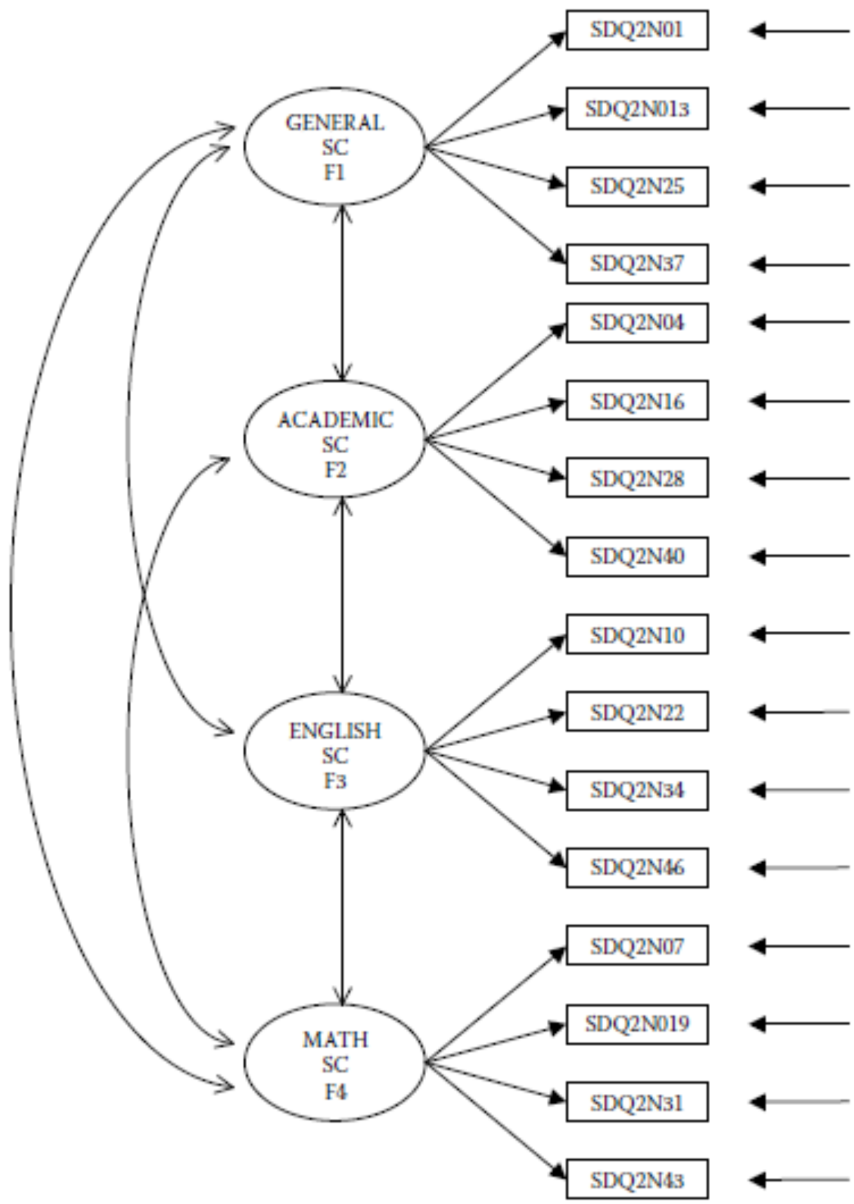


Figure 2: CFA model of self-concept.

a. Using the data that are stored in the dataset `ASC7INDM.csv` with $N = 265$, estimate this model and evaluate its fit using the MFTS statistic reported by `lavaan`. Use the marker variable approach to identify the scale of the latent variables.

Set the working directory to the location where your data file has been downloaded and load the data.

```
# For example.
setwd("/Users/mihai/Downloads")
```

```r
# Load data.
data <- read.csv("ASC7INDM.csv")

# Inspect the data.
View(data)

# Or quickly list the variables.
str(data)

# Or summarize the data.
summary(data)
```

Optional. When working with new datasets, it can also help to have a bird's eye view on the correlation structure of the data. We can use the `corrplot` package to obtain such a plot, which we can install and load as follows:

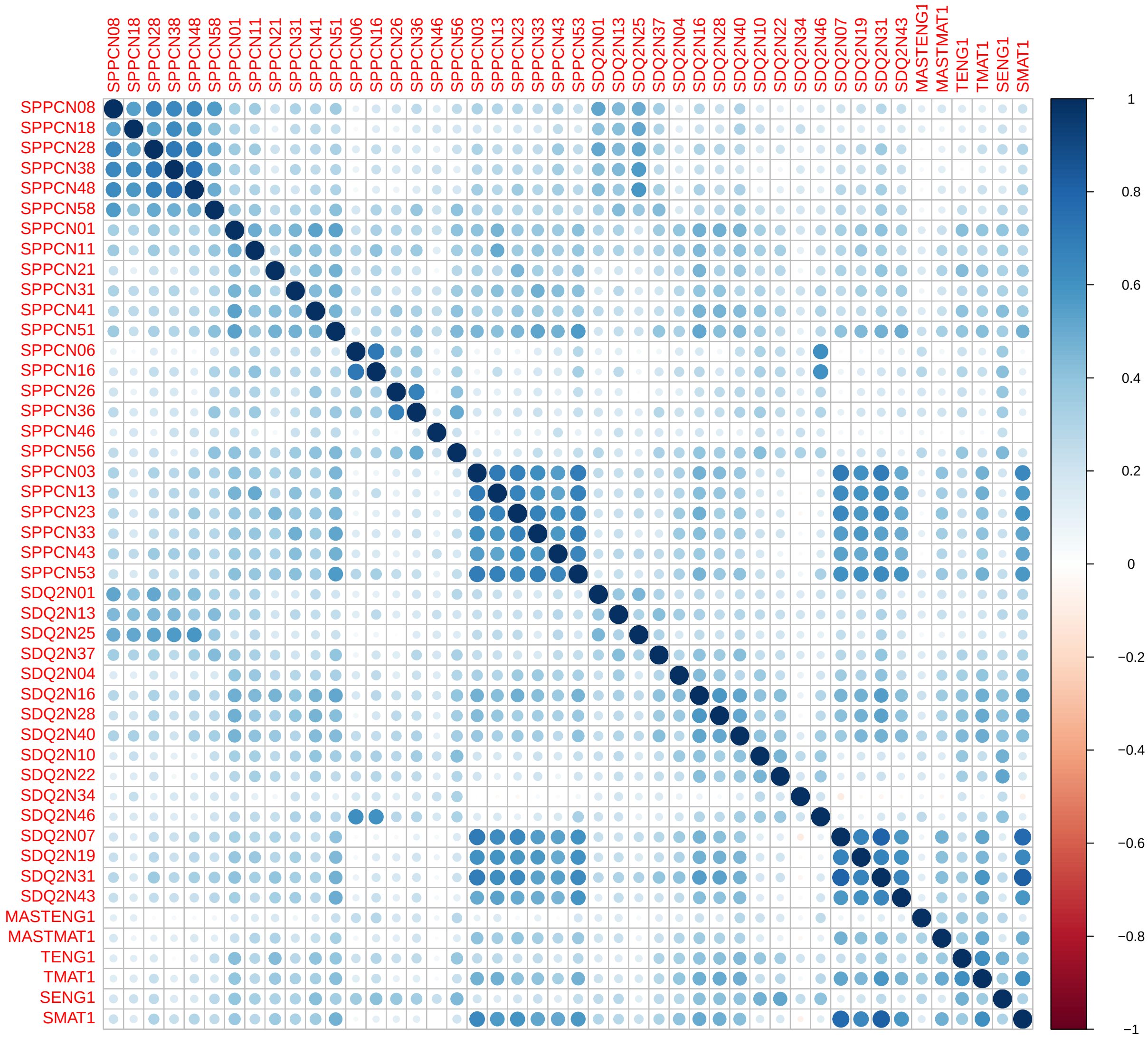
```r
# Install the `corrplot` package.
install.packages("corrplot")

# Load the package.
library(corrplot)
```

Now, we can obtain our correlation plot as follows:

```r
# Compute correlations between all pair of variables.
corrs <- cor(data)

# Plot the correlations.
# Tip: make sure you open your plot in a new window to get a better view.
corrplot(corrs)
```

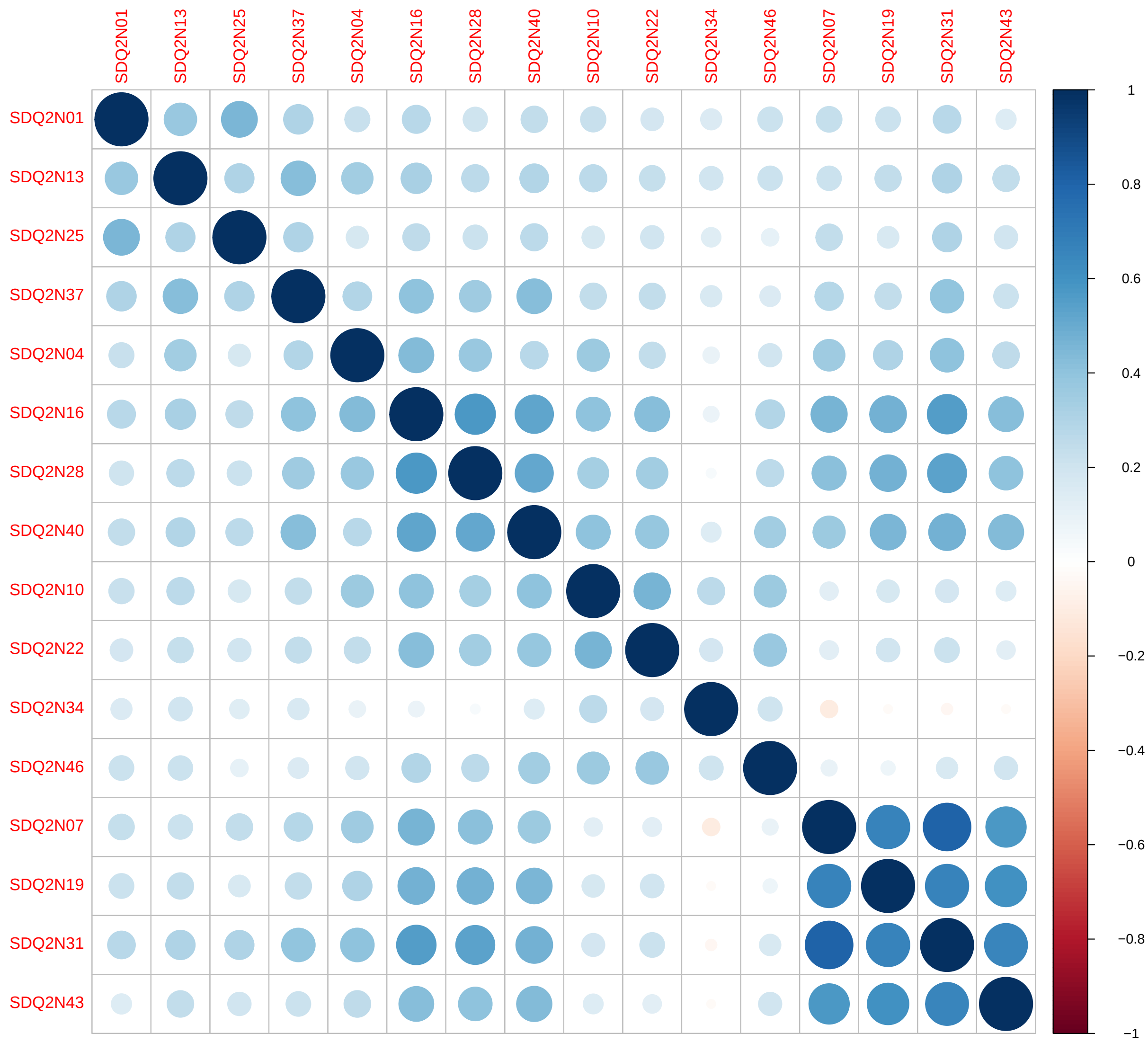


Or, we can obtain the same correlation plot just for the variables we are interested in (i.e., the indicator variables depicted in Figure 2).

```
# Write down the names of the variables we are interested in.
names <- c(
    "SDQ2N01", "SDQ2N13", "SDQ2N25", "SDQ2N37",
    "SDQ2N04", "SDQ2N16", "SDQ2N28", "SDQ2N40",
    "SDQ2N10", "SDQ2N22", "SDQ2N34", "SDQ2N46",
    "SDQ2N07", "SDQ2N19", "SDQ2N31", "SDQ2N43"
)

# Compute the correlations only for the variables listed in `names`.
corrs_variables <- cor(data[, names])

# Plot the correlations.
corrplot(corrs_variables)
```



Now that we've loaded and inspected the data, we can continue with fitting the model depicted in Figure 2. Note that for identification purposes we will use the marker variable approach.

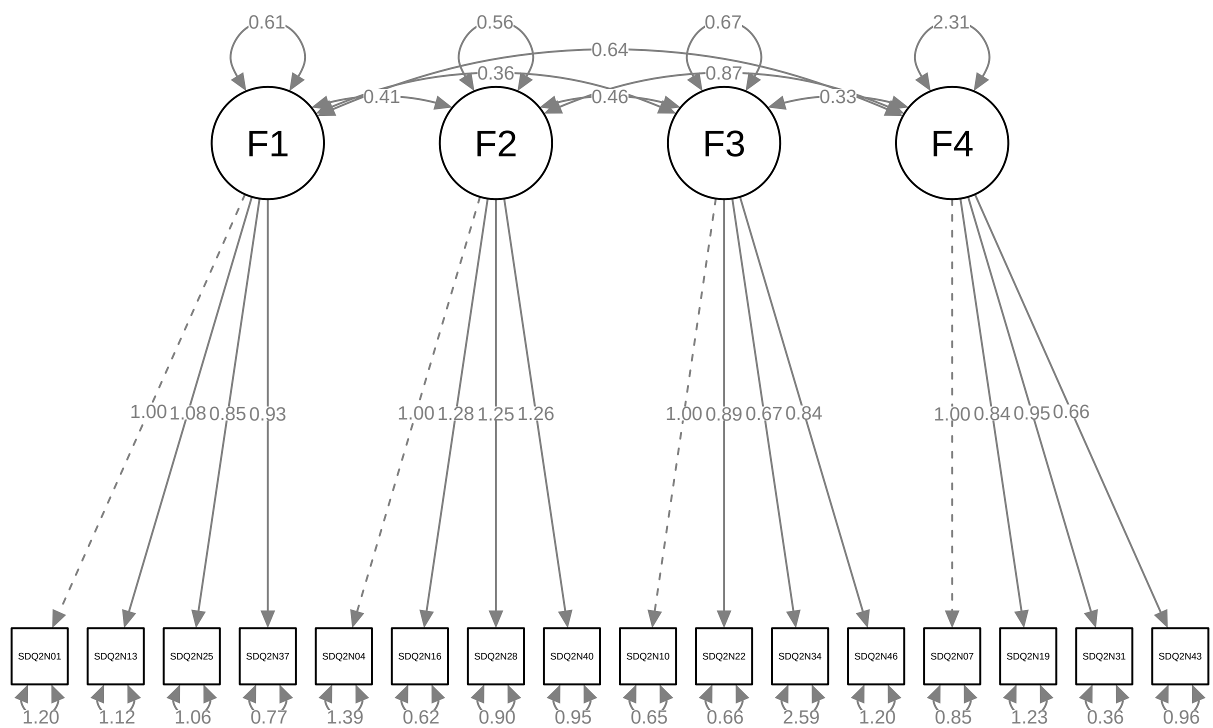
```
# Model syntax.
model_self_concept <- "
    # Measurement model.
    F1 =~ SDQ2N01 + SDQ2N13 + SDQ2N25 + SDQ2N37
    F2 =~ SDQ2N04 + SDQ2N16 + SDQ2N28 + SDQ2N40
    F3 =~ SDQ2N10 + SDQ2N22 + SDQ2N34 + SDQ2N46
    F4 =~ SDQ2N07 + SDQ2N19 + SDQ2N31 + SDQ2N43

    # Covariances between latent variables.
    F1 ~~ F2
    F1 ~~ F3
    F1 ~~ F4
    F2 ~~ F3
    F2 ~~ F4
    F3 ~~ F4
```

```r
"

# Model fit.
model_self_concept_fit <- cfa(model_self_concept, data = data)

# Visualize the model.
semPaths(model_self_concept_fit, what = "paths", whatLabels = "est", sizeMan = 4)
```




```r
# Model summary.
summary(model_self_concept_fit)
```

```
## lavaan 0.6-12 ended normally after 49 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                        38
##
##   Number of observations                           265
##
## Model Test User Model:
##
##   Test statistic                               159.112
##   Degrees of freedom                                98
##   P-value (Chi-square)                           0.000
##
## Parameter Estimates:
##
##   Standard errors                             Standard
##   Information                                 Expected
##   Information saturated (h1) model          Structured
```

```
##
## Latent Variables:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   F1 =~
##     SDQ2N01           1.000
##     SDQ2N13           1.083    0.154    7.044    0.000
##     SDQ2N25           0.851    0.132    6.455    0.000
##     SDQ2N37           0.934    0.131    7.131    0.000
##   F2 =~
##     SDQ2N04           1.000
##     SDQ2N16           1.279    0.150    8.520    0.000
##     SDQ2N28           1.247    0.154    8.097    0.000
##     SDQ2N40           1.259    0.156    8.048    0.000
##   F3 =~
##     SDQ2N10           1.000
##     SDQ2N22           0.889    0.103    8.658    0.000
##     SDQ2N34           0.670    0.148    4.539    0.000
##     SDQ2N46           0.843    0.117    7.225    0.000
##   F4 =~
##     SDQ2N07           1.000
##     SDQ2N19           0.841    0.058   14.495    0.000
##     SDQ2N31           0.952    0.049   19.516    0.000
##     SDQ2N43           0.655    0.049   13.298    0.000
##
## Covariances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   F1 ~~
##     F2                0.415    0.078    5.292    0.000
##     F3                0.355    0.072    4.947    0.000
##     F4                0.635    0.118    5.387    0.000
##   F2 ~~
##     F3                0.464    0.078    5.921    0.000
##     F4                0.873    0.134    6.519    0.000
##   F3 ~~
##     F4                0.331    0.100    3.309    0.001
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##    .SDQ2N01           1.198    0.126    9.537    0.000
##    .SDQ2N13           1.119    0.124    9.019    0.000
##    .SDQ2N25           1.056    0.107    9.897    0.000
##    .SDQ2N37           0.771    0.087    8.821    0.000
##    .SDQ2N04           1.394    0.128   10.900    0.000
##    .SDQ2N16           0.616    0.068    9.020    0.000
##    .SDQ2N28           0.896    0.090    9.959    0.000
##    .SDQ2N40           0.952    0.095   10.029    0.000
##    .SDQ2N10           0.653    0.082    7.941    0.000
##    .SDQ2N22           0.657    0.075    8.735    0.000
##    .SDQ2N34           2.590    0.233   11.128    0.000
##    .SDQ2N46           1.201    0.118   10.183    0.000
##    .SDQ2N07           0.854    0.100    8.551    0.000
```

```
##    .SDQ2N19          1.228    0.121   10.153    0.000
##    .SDQ2N31          0.365    0.065    5.649    0.000
##    .SDQ2N43          0.964    0.092   10.473    0.000
##     F1               0.613    0.137    4.464    0.000
##     F2               0.561    0.126    4.453    0.000
##     F3               0.668    0.116    5.749    0.000
##     F4               2.307    0.273    8.460    0.000
```

We obtain a $\chi^2 = 159.112$ with $DF = 98$ and a p-value < 0.001. The *null hypothesis* that the model implied covariance matrix fits the population covariance matrix must be rejected. You do not want to reject the null hypothesis (i.e., you want a p-value above a certain *alpha*).

Note. The covariances between the latent variables are estimated by default when using `lavaan`. These covariances were added in the syntax above just to illustrate how this is done.

b. According to the `lavaan` results, this model has 98 degrees of freedom. Show calculations that clarify why this model has 98 degrees of freedom.

The variance-covariance matrix Σ has $\frac{16\times(16+1)}{2} = 136$ elements. In the model we have 38 parameters:

- 16 variances of error terms
- 4 variances of latent variables
- 6 covariances
- 12 loadings (i.e., not 16 because we implement 4 marker constraints)

We compute the degrees of freedom as

$$DF = \#\text{ parameters} - \#\text{ free parameters}$$

and obtain $136 - 38 = 98$.

c. Which possibilities do you have to possibly improve the fit of the model?

Ways to possibly improve model fit:

- include cross-loadings
- include error covariances,
- constrain parameters to certain values

We may also add equality constraints, or constrain non-significant loadings and/ or covariances to 0. However, this will not improve the fit, it will only ensure that the model fit will not become worse.

References

Umstattd Meyer, M. R., Janke, M. C., & Beaujean, A. A. (2014). Predictors of Older Adults' Personal and Community Mobility: Using a Comprehensive Theoretical Mobility Framework. *The Gerontologist*, *54*(3), 398–408. https://doi.org/10.1093/geront/gnt054