title: “Assignment 2: Task 1 - CFA” author: “Group 11” date: “30.10.2018” output: pdf_document —

0. Load Libraries

```
library(tidyverse)
library(lavaan)
library(semPlot)
library(lessR)
```

1. Problem Statement

We are given a covariance matrix and other descriptive statistics (mean, standard deviation, number of observations) of 8 variables that are assumed to measure abilities of professional basketball players. The task is to check whether a hypothesized model with two latent factors called “frontcourt skills” and “backcourt skills” fits the variance/covariance structure of the given data well, compare this model to another with three latent factors and finally perform a test for equality of two parameters. The goal is to find shared structures in the variances of the variables, i.e. factors that have an influence on groups of variables and can therefore be determined underlying unobservable factors that help explaining the variance/covariance structure of the data.

2. Descriptive Statistics

With only the variance matrix there is no possibility of checking for outliers in the data. One can see from looking at the complete table that the means of X3 and X8 and are substantially higher than the rest. This is not important for the analysis and can be safely ignored.

```
basketdf <- read.table("http://feb.kuleuven.be/martina.vandebroek/public/STATdata/basketball.txt", heade
basket_cov <- basketdf %>% filter(X_TYPE_ == "COV") %>% select(2:ncol(basketdf))
rownames(basket_cov) <- colnames(basket_cov)
basket_cov <- as.matrix(basket_cov)
basketdf
```

```
##    X_TYPE_         X1         X2         X3         X4         X5
## 1     MEAN   0.313544   0.757588   3.354687   1.137500   0.508394
## 2      STD   0.126765   0.101748   2.051235   0.420963   0.063846
## 3        N 320.000000 320.000000 320.000000 320.000000 320.000000
## 4      COV   0.016069   0.006183   0.033464   0.002353  -0.002755
## 5      COV   0.006183   0.010353   0.034056   0.001675  -0.001772
## 6      COV   0.033464   0.034056   4.207564   0.343115  -0.025018
## 7      COV   0.002353   0.001675   0.343115   0.177210  -0.001377
## 8      COV  -0.002755  -0.001772  -0.025018  -0.001377   0.004076
## 9      COV  -0.030210  -0.024046  -0.331983  -0.011191   0.019510
## 10     COV  -0.086770  -0.058966  -0.775118  -0.090470   0.041064
## 11     COV  -0.091280  -0.066631  -0.741020  -0.131826   0.055360
##           X6         X7         X8
## 1   0.755000   1.495000   5.056875
## 2   0.655414   1.178294   2.068820
## 3 320.000000 320.000000 320.000000
## 4  -0.030210  -0.086770  -0.091280
```

```
## 5   -0.024046  -0.058966  -0.066631
## 6   -0.331983  -0.775118  -0.741020
## 7   -0.011191  -0.090470  -0.131826
## 8    0.019510   0.041064   0.055360
## 9    0.429567   0.472063   0.775828
## 10   0.472063   1.388376   1.795144
## 11   0.775828   1.795144   4.280015
```

3. Assumptions

The only necessary condition to obtain meaningful results is that the number of inputs (unique values in variance/covariance matrix) is higher than the number of estimated parameters. Here we have 8 variables so our number of inputs is (8*(8+1))/2=36. For the first model we estimate 8 loadings, 8 error variances of the variables and 1 covariance between factors which lead to a total of 17 estimated parameters. The model is therefore well identified (as is the second model where 19 parameters are estimated). We confirm this with the inspect function where nonzero integers in the output are parameters that are to be estimated.

```r
# Specify two-factor model
model1 <- '
# latent variables
backcourt =~ X1 + X2 + X3 + X4
frontcourt =~ X5 + X6 + X7 + X8
'
fit1 <- lavaan::sem(model1, sample.nobs = 320, sample.cov = basket_cov, orthogonal = F)
#number of estimated parameters
inspect(fit1)
```

```
## $lambda
##    bckcrt frntcr
## X1      0      0
## X2      1      0
## X3      2      0
## X4      3      0
## X5      0      0
## X6      0      4
## X7      0      5
## X8      0      6
##
## $theta
##    X1 X2 X3 X4 X5 X6 X7 X8
## X1  7
## X2  0  8
## X3  0  0  9
## X4  0  0  0 10
## X5  0  0  0  0 11
## X6  0  0  0  0  0 12
## X7  0  0  0  0  0  0 13
## X8  0  0  0  0  0  0  0 14
##
## $psi
##            bckcrt frntcr
## backcourt  15
## frontcourt 17     16
```

```r
inspect(fit1,"est")
```

```
## $lambda
##    bckcrt frntcr
## X1  1.000  0.000
## X2  0.710  0.000
## X3  7.821  0.000
## X4  0.897  0.000
## X5  0.000  1.000
## X6  0.000 11.683
## X7  0.000 31.256
## X8  0.000 43.105
##
## $theta
##    X1    X2    X3    X4    X5    X6    X7    X8
## X1 0.008
## X2 0.000 0.006
## X3 0.000 0.000 3.729
## X4 0.000 0.000 0.000 0.171
## X5 0.000 0.000 0.000 0.000 0.003
## X6 0.000 0.000 0.000 0.000 0.000 0.248
## X7 0.000 0.000 0.000 0.000 0.000 0.000 0.092
## X8 0.000 0.000 0.000 0.000 0.000 0.000 0.000 1.810
##
## $psi
##            bckcrt frntcr
## backcourt   0.008
## frontcourt -0.003  0.001
```

4. Method and interpretation

Part A:

The model was already fit in section 3 with the sem function from the lavaan package. To assess the model fit we constructed a function that computes the goodness-of-fit (GFI) because this indicator is not given in the output of lavaan.

```r
GFI <- function(Si, Sobs){
  if( class(Si) == "list"){
    Si <- as.matrix(as.data.frame(Si$cov))
  }

  nominator <- sum(diag(solve(Si) %*% Sobs - diag(ncol(Sobs))))^2
  denominator <- sum(diag(solve(Si) %*% Sobs))^2
  return(1 - (nominator / denominator))
}
```

By using the summary, resid and modindices functions of the lavaan package the model is inspected. Overall the model does not fit the data well, for example the null hypothesis that the estimated covariance matrix is equal to the observed covariance matrix is rejected with a p-value of 0.

```r
# Analysis of model fit
summary(fit1,fit.measures=TRUE)
```

```
## lavaan 0.6-3 ended normally after 88 iterations
##
##   Optimization method                           NLMINB
##   Number of free parameters                         17
##
##   Number of observations                           320
##
##   Estimator                                         ML
##   Model Fit Test Statistic                     113.897
##   Degrees of freedom                                19
##   P-value (Chi-square)                           0.000
##
## Model test baseline model:
##
##   Minimum Function Test Statistic              909.437
##   Degrees of freedom                                28
##   P-value                                        0.000
##
## User model versus baseline model:
##
##   Comparative Fit Index (CFI)                    0.892
##   Tucker-Lewis Index (TLI)                       0.841
##
## Loglikelihood and Information Criteria:
##
##   Loglikelihood user model (H0)              -1061.042
##   Loglikelihood unrestricted model (H1)      -1004.093
##
##   Number of free parameters                         17
##   Akaike (AIC)                                2156.084
##   Bayesian (BIC)                              2220.145
##   Sample-size adjusted Bayesian (BIC)         2166.224
##
## Root Mean Square Error of Approximation:
##
##   RMSEA                                          0.125
##   90 Percent Confidence Interval          0.103  0.148
##   P-value RMSEA <= 0.05                          0.000
##
## Standardized Root Mean Square Residual:
##
##   SRMR                                           0.073
##
## Parameter Estimates:
##
##   Information                                 Expected
##   Information saturated (h1) model          Structured
##   Standard Errors                             Standard
##
## Latent Variables:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   backcourt =~
##     X1                1.000
##     X2                0.710    0.079    8.985    0.000
```

```
##     X3                7.821    1.507    5.189    0.000
##     X4                0.897    0.304    2.947    0.003
##   frontcourt =~
##     X5                1.000
##     X6               11.683    1.249    9.351    0.000
##     X7               31.256    2.765   11.305    0.000
##     X8               43.105    4.165   10.349    0.000
##
## Covariances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   backcourt ~~
##     frontcourt       -0.003    0.000   -7.107    0.000
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##    .X1                0.008    0.001    8.842    0.000
##    .X2                0.006    0.001   10.384    0.000
##    .X3                3.729    0.305   12.228    0.000
##    .X4                0.171    0.014   12.531    0.000
##    .X5                0.003    0.000   12.195    0.000
##    .X6                0.248    0.021   11.884    0.000
##    .X7                0.092    0.042    2.205    0.027
##    .X8                1.810    0.167   10.869    0.000
##     backcourt         0.008    0.001    5.970    0.000
##     frontcourt        0.001    0.000    5.407    0.000
```

```r
GFI(Si = fitted(fit1), Sobs = basket_cov)
```

```
## [1] 0.9999902
```

```r
# GFI > 0.95 --> good
# Baseline model and specified model p-value < 5% --> bad
# Comparative Fit index 0.892 < 0.95 --> bad
# --> overall bad model fit

resid(fit1, type = "standardized")
```

```
## $type
## [1] "standardized"
##
## $cov
##    X1     X2     X3     X4     X5     X6     X7     X8
## X1  0.000
## X2  3.282  0.000
## X3 -3.381 -1.090  0.000
## X4 -2.510 -1.897  6.089  0.000
## X5 -0.124  0.643 -0.648  0.859  0.000
## X6  0.583 -0.638 -1.506  1.465  2.924  0.000
## X7 -1.864  1.439 -2.563 -1.476 -1.325 -5.744  0.000
## X8  3.842  2.795  1.109 -0.764 -0.526  3.243  1.592  0.000
```

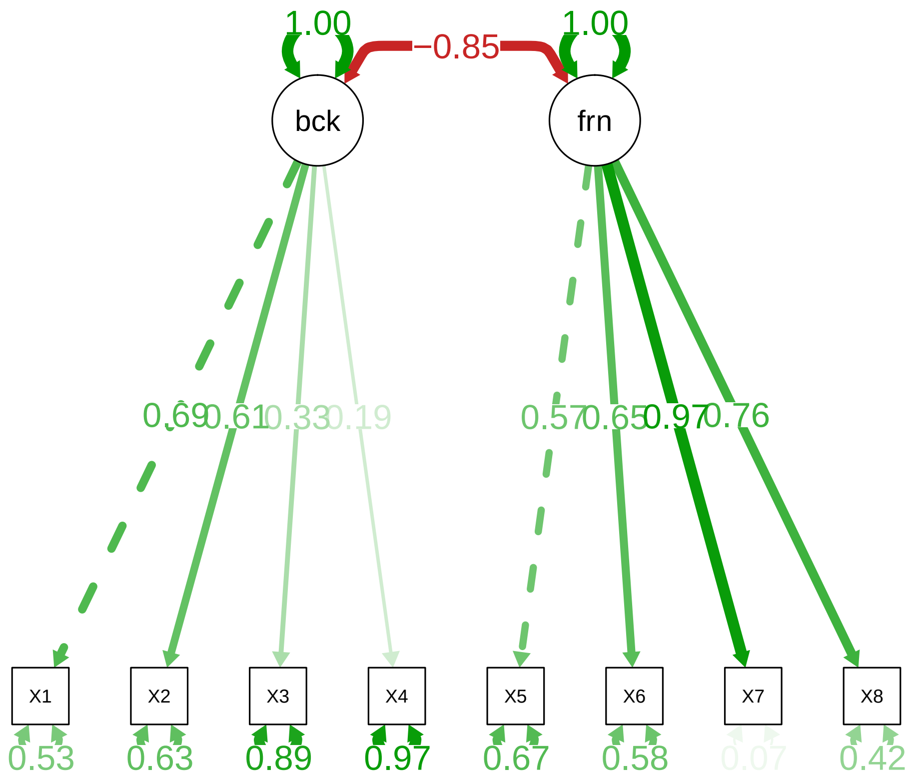
```r
# Largest standardized residuals > 1.96
# X4, X3: 6.089
# X7, X6: -5.744
# X8, X1: 3.842
# X3, X1: -3.381
```

```r
# X2, X1: 3.282
# X8, X6: 3.243
# X6, X5: 2.924
# X8, X2: 2.795
# X7, X3: -2.563

# Calculate modification indices
modindices(fit1, sort. = T)
```

```
##          lhs op rhs     mi     epc sepc.lv sepc.all sepc.nox
## 41        X3 ~~  X4 43.828   0.302   0.302    0.378    0.378
## 53        X6 ~~  X7 23.085  -0.136  -0.136   -0.898   -0.898
## 23 backcourt =~  X8 22.190  17.306   1.510    0.731    0.731
## 22 backcourt =~  X7 18.418 -10.325  -0.901   -0.766   -0.766
## 28        X1 ~~  X2 16.622   0.004   0.004    0.506    0.506
## 54        X6 ~~  X8 11.856   0.154   0.154    0.230    0.230
## 34        X1 ~~  X8 10.642   0.028   0.028    0.223    0.223
## 29        X1 ~~  X3 10.552  -0.040  -0.040   -0.223   -0.223
## 50        X5 ~~  X6  8.661   0.005   0.005    0.176    0.176
## 33        X1 ~~  X7  7.776  -0.013  -0.013   -0.479   -0.479
## 25 frontcourt =~  X2  7.547   2.728   0.099    0.976    0.976
## 30        X1 ~~  X4  6.081  -0.006  -0.006   -0.161   -0.161
## 26 frontcourt =~  X3  5.562 -25.879  -0.941   -0.459   -0.459
## 44        X3 ~~  X7  3.606  -0.140  -0.140   -0.238   -0.238
## 36        X2 ~~  X4  3.541  -0.004  -0.004   -0.116   -0.116
## 47        X4 ~~  X6  3.366   0.022   0.022    0.105    0.105
## 55        X7 ~~  X8  2.999   0.187   0.187    0.458    0.458
## 40        X2 ~~  X8  2.935   0.012   0.012    0.110    0.110
## 45        X3 ~~  X8  2.732   0.255   0.255    0.098    0.098
## 48        X4 ~~  X7  2.446  -0.024  -0.024   -0.189   -0.189
## 51        X5 ~~  X7  1.586  -0.003  -0.003   -0.204   -0.204
## 46        X4 ~~  X5  1.239   0.001   0.001    0.063    0.063
## 27 frontcourt =~  X4  1.228  -2.371  -0.086   -0.205   -0.205
## 35        X2 ~~  X3  1.182  -0.011  -0.011   -0.069   -0.069
## 43        X3 ~~  X6  1.062  -0.057  -0.057   -0.060   -0.060
## 38        X2 ~~  X6  1.047  -0.003  -0.003   -0.063   -0.063
## 32        X1 ~~  X6  0.520   0.002   0.002    0.047    0.047
## 49        X4 ~~  X8  0.510  -0.023  -0.023   -0.042   -0.042
## 52        X5 ~~  X8  0.269  -0.002  -0.002   -0.033   -0.033
## 37        X2 ~~  X5  0.183   0.000   0.000    0.026    0.026
## 24 frontcourt =~  X1  0.080   0.442   0.016    0.127    0.127
## 42        X3 ~~  X5  0.061  -0.001  -0.001   -0.014   -0.014
## 31        X1 ~~  X5  0.057   0.000   0.000   -0.015   -0.015
## 20 backcourt =~  X5  0.051   0.028   0.002    0.039    0.039
## 21 backcourt =~  X6  0.019  -0.171  -0.015   -0.023   -0.023
## 39        X2 ~~  X7  0.002   0.000   0.000    0.006    0.006
```

```r
# Plot using standardized loading estimates
semPaths(fit1, "std",edge.label.cex = 1.4, curvePivot = TRUE)
```



Part B:

The model is respecified by splitting the backcourt factor into 2 factors: “shooting skills” and “neuromuscular coordination” while the frontcourt factor is equivalent to the first model.

```
model2 <- '
# latent variables
shoot =~ X1 + X2
neuro =~ X3 + X4
frontcourt =~ X5 + X6 + X7 + X8
'

# Fitting the model
fit2 <- lavaan::sem(model2, sample.nobs = 320, sample.cov = basket_cov, orthogonal = F)

# Analysis of model fit
summary(fit2,fit.measures=TRUE)
```

```
## lavaan 0.6-3 ended normally after 96 iterations
##
##   Optimization method                           NLMINB
##   Number of free parameters                         19
##
##   Number of observations                           320
##
##   Estimator                                         ML
##   Model Fit Test Statistic                      54.516
```

```
##   Degrees of freedom                                17
##   P-value (Chi-square)                           0.000
##
## Model test baseline model:
##
##   Minimum Function Test Statistic              909.437
##   Degrees of freedom                                28
##   P-value                                        0.000
##
## User model versus baseline model:
##
##   Comparative Fit Index (CFI)                    0.957
##   Tucker-Lewis Index (TLI)                       0.930
##
## Loglikelihood and Information Criteria:
##
##   Loglikelihood user model (H0)              -1031.351
##   Loglikelihood unrestricted model (H1)      -1004.093
##
##   Number of free parameters                         19
##   Akaike (AIC)                                2100.703
##   Bayesian (BIC)                              2172.301
##   Sample-size adjusted Bayesian (BIC)         2112.036
##
## Root Mean Square Error of Approximation:
##
##   RMSEA                                          0.083
##   90 Percent Confidence Interval          0.059  0.108
##   P-value RMSEA <= 0.05                          0.014
##
## Standardized Root Mean Square Residual:
##
##   SRMR                                           0.038
##
## Parameter Estimates:
##
##   Information                                 Expected
##   Information saturated (h1) model          Structured
##   Standard Errors                             Standard
##
## Latent Variables:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   shoot =~
##     X1                1.000
##     X2                0.681    0.074    9.157    0.000
##   neuro =~
##     X3                1.000
##     X4                0.120    0.034    3.569    0.000
##   frontcourt =~
##     X5                1.000
##     X6               11.675    1.250    9.342    0.000
##     X7               31.338    2.771   11.310    0.000
##     X8               43.074    4.167   10.338    0.000
##
```

```
## Covariances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   shoot ~~
##     neuro              0.037    0.013    2.739    0.006
##     frontcourt        -0.003    0.000   -7.099    0.000
##   neuro ~~
##     frontcourt        -0.024    0.005   -4.889    0.000
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##    .X1                 0.007    0.001    6.979    0.000
##    .X2                 0.006    0.001    9.885    0.000
##    .X3                 1.347    0.769    1.750    0.080
##    .X4                 0.136    0.015    8.831    0.000
##    .X5                 0.003    0.000   12.208    0.000
##    .X6                 0.248    0.021   11.906    0.000
##    .X7                 0.089    0.042    2.127    0.033
##    .X8                 1.819    0.167   10.918    0.000
##     shoot              0.009    0.001    6.408    0.000
##     neuro              2.848    0.824    3.455    0.001
##     frontcourt         0.001    0.000    5.404    0.000
```

```
GFI(Si = fitted(fit2), Sobs = basket_cov)
```

```
## [1] 0.9999902
```

```
# GFI > 0.95 --> good
# Baseline model and specified model p-value < 5% --> bad
# Comparative Fit index 0.975 > 0.95 --> good
# AIC: 2100.703 < 2156.084 --> model2 better
# BIC: 2172.301 < 2220.145 --> model2 better

resid(fit2, type = "standardized")
```

```
## $type
## [1] "standardized"
##
## $cov
##    X1     X2     X3     X4     X5     X6     X7     X8
## X1  0.000
## X2  0.000  0.000
## X3 -0.553  1.247  0.000
## X4 -0.902 -0.666  0.000  0.000
## X5 -0.131  0.316 -0.154  1.266  0.000
## X6  0.566 -1.047 -0.949  1.981  2.952  0.000
## X7 -2.386 -1.501 -1.286  0.092 -1.439 -5.924  0.000
## X8  3.867  2.249  2.218 -0.234 -0.474  3.280  1.740  0.000
```

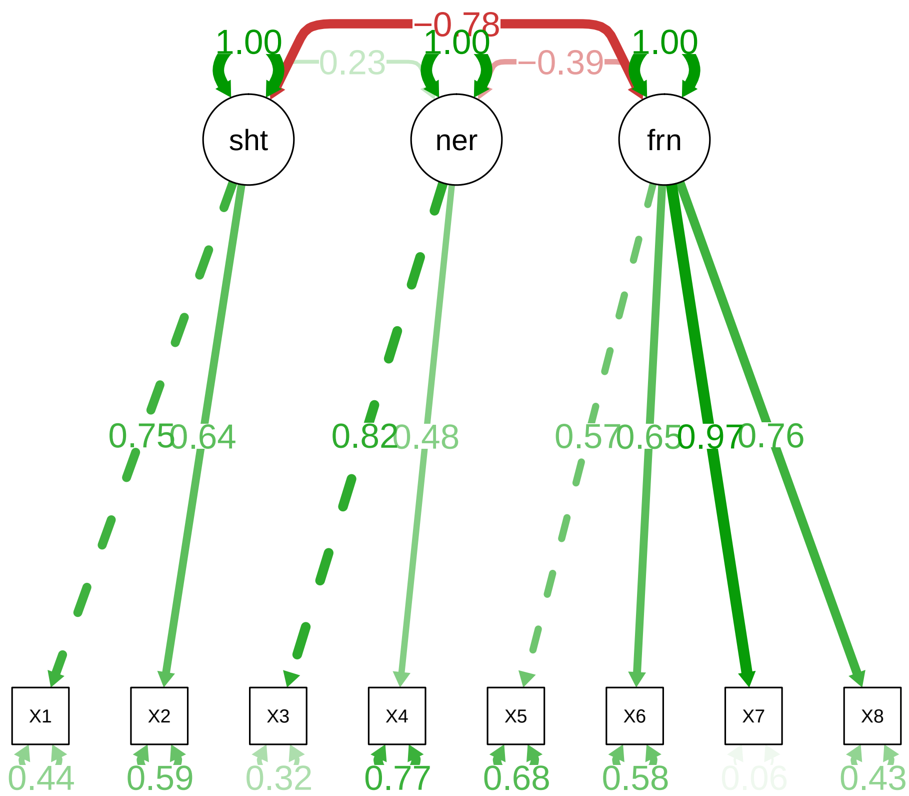
```
# Largest standardized residuals > 1.96
# X7, X6: -5.924
# X8, X1: 3.867
# X8, X6: 3.280
# X6, X5: 2.952
# X7, X1: -2.386
# X8, X2: 2.249
# X8, X3: 2.218
```

```r
modindices(fit2, sort. = T)
```

```
##             lhs op rhs     mi     epc sepc.lv sepc.all sepc.nox
## 64           X6 ~~  X7 24.343  -0.139  -0.139   -0.936   -0.936
## 28        shoot =~  X8 14.341   8.533   0.812    0.393    0.393
## 65           X6 ~~  X8 12.153   0.156   0.156    0.232    0.232
## 45           X1 ~~  X8 10.823   0.027   0.027    0.242    0.242
## 27        shoot =~  X7 10.078  -4.389  -0.417   -0.355   -0.355
## 44           X1 ~~  X7  9.638  -0.015  -0.015   -0.599   -0.599
## 61           X5 ~~  X6  8.788   0.005   0.005    0.177    0.177
## 56           X3 ~~  X8  7.846   0.405   0.405    0.258    0.258
## 58           X4 ~~  X6  5.466   0.026   0.026    0.140    0.140
## 66           X7 ~~  X8  3.629   0.205   0.205    0.510    0.510
## 34        neuro =~  X8  3.039   0.109   0.185    0.089    0.089
## 54           X3 ~~  X6  2.929  -0.089  -0.089   -0.154   -0.154
## 51           X2 ~~  X8  1.905   0.009   0.009    0.090    0.090
## 62           X5 ~~  X7  1.863  -0.004  -0.004   -0.225   -0.225
## 46           X2 ~~  X3  1.849   0.013   0.013    0.141    0.141
## 59           X4 ~~  X7  1.684  -0.020  -0.020   -0.187   -0.187
## 57           X4 ~~  X5  1.644   0.001   0.001    0.076    0.076
## 60           X4 ~~  X8  1.595  -0.038  -0.038   -0.077   -0.077
## 33        neuro =~  X7  1.405  -0.041  -0.070   -0.059   -0.059
## 49           X2 ~~  X6  1.359  -0.003  -0.003   -0.073   -0.073
## 24        shoot =~  X4  1.051  -0.425  -0.040   -0.096   -0.096
## 38 frontcourt =~  X4  1.051   2.984   0.108    0.258    0.258
## 23        shoot =~  X3  1.051   3.535   0.336    0.164    0.164
## 37 frontcourt =~  X3  1.051 -24.841  -0.902   -0.441   -0.441
## 36 frontcourt =~  X2  0.888  -1.557  -0.057   -0.557   -0.557
## 29        neuro =~  X1  0.888  -0.006  -0.010   -0.075   -0.075
## 35 frontcourt =~  X1  0.888   2.285   0.083    0.656    0.656
## 30        neuro =~  X2  0.888   0.004   0.006    0.064    0.064
## 47           X2 ~~  X4  0.665  -0.002  -0.002   -0.052   -0.052
## 43           X1 ~~  X6  0.629   0.002   0.002    0.055    0.055
## 55           X3 ~~  X7  0.406  -0.050  -0.050   -0.146   -0.146
## 53           X3 ~~  X5  0.331  -0.003  -0.003   -0.051   -0.051
## 40           X1 ~~  X3  0.230  -0.006  -0.006   -0.062   -0.062
## 63           X5 ~~  X8  0.218  -0.002  -0.002   -0.030   -0.030
## 41           X1 ~~  X4  0.194  -0.001  -0.001   -0.031   -0.031
## 50           X2 ~~  X7  0.185  -0.002  -0.002   -0.071   -0.071
## 48           X2 ~~  X5  0.175   0.000   0.000    0.026    0.026
## 32        neuro =~  X6  0.148  -0.009  -0.015   -0.022   -0.022
## 42           X1 ~~  X5  0.066   0.000   0.000   -0.018   -0.018
## 31        neuro =~  X5  0.023   0.000   0.001    0.009    0.009
## 26        shoot =~  X6  0.004  -0.051  -0.005   -0.007   -0.007
## 25        shoot =~  X5  0.002   0.004   0.000    0.005    0.005
```

```r
# Maybe introduce forth factor containing X6, X7

# Plot using standardized loading estimates
semPaths(fit2, "std",edge.label.cex = 1.4, curvePivot = TRUE)
```



The second model has a higher GFI, both a lower AIC and BIC but still a p-value of 0. It is thus better than the first model but still not a very good fit to the data. The model could probably be improved by introducing new factor for X6 and X7 and/or a loading of the factor shooting skills on X8 (these two have the highest mi value in the modindices output).

Part C:

In order to be able to test whether the loadings of X7 and x8 are the same, a two sided t-test using the standardized loadings needs to be applied. The null hypothesis is that the loadings are equal, whereas the alternative hypothesis is that both values are unequal. The significance level is set to five percent and the remaining degrees of freedom are 301, as 19 variables were estimated.

```
# Test whether loadings of X7 and X8 are the same (t-Test)
estX7X8 <- standardizedSolution(fit2)$est.std[7:8] # estimates
seX7X8 <- standardizedSolution(fit2)$se[7:8] # standard errors

# H0: X7 = X8
# H1: X7 != X8
alpha <- 0.05

# One of the folliwing lines has to be TRUE in order to reject H0
((estX7X8[1] - estX7X8[2]) / seX7X8[1]) > qt(p=1-(alpha/2), df=320-19)
```

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

```
# OR
((estX7X8[1] - estX7X8[2]) / seX7X8[1]) < -qt(p=1-(alpha/2), df=320-19)
```

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

The resulting test statistic is equal to 13.34606, which is much larger then the threshold value of approximately 1.94. Therefore the null hypothesis can be rejected, meaning that the loadings are significantly different from each other.

5. Alternative solutions

??Maybe specify another model??

6. Conclusion

???