Bayesian CFA & SEM

José G. Dias

Table of contents

1. Summary

This text explores the concepts of Confirmatory Factorial Analysis (CFA) and Structural Equation Models (SEM) in the Bayesian framework. It is mostly developed using the lavaan syntax for SEM specification of the model and the estimation by blavaan.

2. Data set

The data set to be used in the modeling contains 120 observations and 6 variables:

```
# Data
dat <- read.csv("pos_neg.csv")
dim(dat)
```

```
[1] 120   6
```

```
summary(dat)
```

```
     great          cheerful          happy             sad         
 Min.   :1.021   Min.   :1.114   Min.   :1.216   Min.   :-0.4846  
 1st Qu.:2.467   1st Qu.:2.383   1st Qu.:2.569   1st Qu.: 1.4784  
 Median :2.973   Median :2.970   Median :3.119   Median : 1.9434  
 Mean   :2.979   Mean   :2.952   Mean   :3.111   Mean   : 1.9659  
 3rd Qu.:3.502   3rd Qu.:3.498   3rd Qu.:3.699   3rd Qu.: 2.5105  
 Max.   :5.312   Max.   :5.761   Max.   :5.066   Max.   : 3.8678  
      down            unhappy        
 Min.   :-0.1142   Min.   :-0.04151  
 1st Qu.: 1.1217   1st Qu.: 1.25618  
 Median : 1.5825   Median : 1.74239  
 Mean   : 1.5934   Mean   : 1.68107  
 3rd Qu.: 2.0512   3rd Qu.: 2.14718  
 Max.   : 3.5776   Max.   : 3.08815  
```

It contains three positive feeling and 3 negative feelings. Thus, we can hypothesize that the first 3 indicators - great, cheerful and happy - measure positive feelings, whereas the last three - sad, down and unhappy - are negative feelings.

3. Our first CFA model

3.1 Model specification

As three feelings are positive (great, cheerful and happy), we can assume a common factor called Positive:

$$great_i = \mu_1 + \lambda_1 Positive_i + \epsilon_{1i}$$

$$cheerful_i = \mu_2 + \lambda_2 Positive_i + \epsilon_{2i}$$

$$happy_i = \mu_3 + \lambda_3 Positive_i + \epsilon_{3i}$$

And the same definition for the Negative feelings:

$$sad_i = \mu_4 + \lambda_4 Negative_i + \epsilon_{4i}$$

$$down_i = \mu_5 + \lambda_5 Negative_i + \epsilon_{5i}$$

$$unhappy_i = \mu_6 + \lambda_6 Negative_i + \epsilon_{6i}$$

The Lavaan syntax is:

```
model.cfa1 <- 'Positive =~ great + cheerful + happy
               Negative =~ sad   + down     + unhappy'
```

3.2 Model estimation

Let us install and load the needed packages:

```
#install.packages("rstan")
#install.packages("quadprog")
#install.packages("pbivnorm")
#install.packages("CompQuadForm")
#install.packages("mvtnorm")
#install.packages("sandwich")
#install.packages("future")
#install.packages("backports")
#install.packages("brms")
#install.packages("psych")

library(rstan)
```

```
Loading required package: StanHeaders


rstan version 2.32.6 (Stan version 2.32.2)

For execution on a local, multicore CPU with excess RAM we recommend calling
options(mc.cores = parallel::detectCores()).
To avoid recompilation of unchanged Stan programs, we recommend calling
rstan_options(auto_write = TRUE)
```

```
For within-chain threading using `reduce_sum()` or `map_rect()` Stan
functions,
change `threads_per_chain` option:
rstan_options(threads_per_chain = 1)

Do not specify '-march=native' in 'LOCAL_CPPFLAGS' or a Makevars file
```

```r
library(lavaan)
```

```
This is lavaan 0.6-17
lavaan is FREE software! Please report any bugs.
```

```r
library(blavaan)
```

```
Loading required package: Rcpp

This is blavaan 0.5-4

On multicore systems, we suggest use of future::plan("multicore") or
  future::plan("multisession") for faster post-MCMC computations.
```

```r
library(brms)
```

```
Loading 'brms' package (version 2.21.0). Useful instructions
can be found by typing help('brms'). A more detailed introduction
to the package is available through vignette('brms_overview').


Attaching package: 'brms'

The following object is masked from 'package:rstan':

    loo

The following object is masked from 'package:stats':

    ar
```

```r
library(psych)
```

```
Attaching package: 'psych'

The following object is masked from 'package:brms':

    cs

The following object is masked from 'package:lavaan':

    cor2cov
```

```
The following object is masked from 'package:rstan':

    lookup
```

```r
library(bayesplot)
```

```
This is bayesplot version 1.11.1

- Online documentation and vignettes at mc-stan.org/bayesplot

- bayesplot theme set to bayesplot::theme_default()

   * Does _not_ affect other ggplot2 plots

   * See ?bayesplot_theme_set for details on theme setting


Attaching package: 'bayesplot'

The following object is masked from 'package:brms':

    rhat
```

```r
rstan_options(auto_write = TRUE)
options(mc.cores = parallel::detectCores())
```

Now we can run the first model:

```r
modelfit.cfa1 <-  bcfa(model.cfa1, data=dat, std.lv=T,
                       n.chains = 3, burnin=5000,
                       sample=1000, target = "stan")
```

```
Computing post-estimation metrics (including lvs if requested)...
```

The estimates are:

```r
summary(modelfit.cfa1, standardized=T,
        rsquare=T, neff=TRUE, postmedian=T)
```

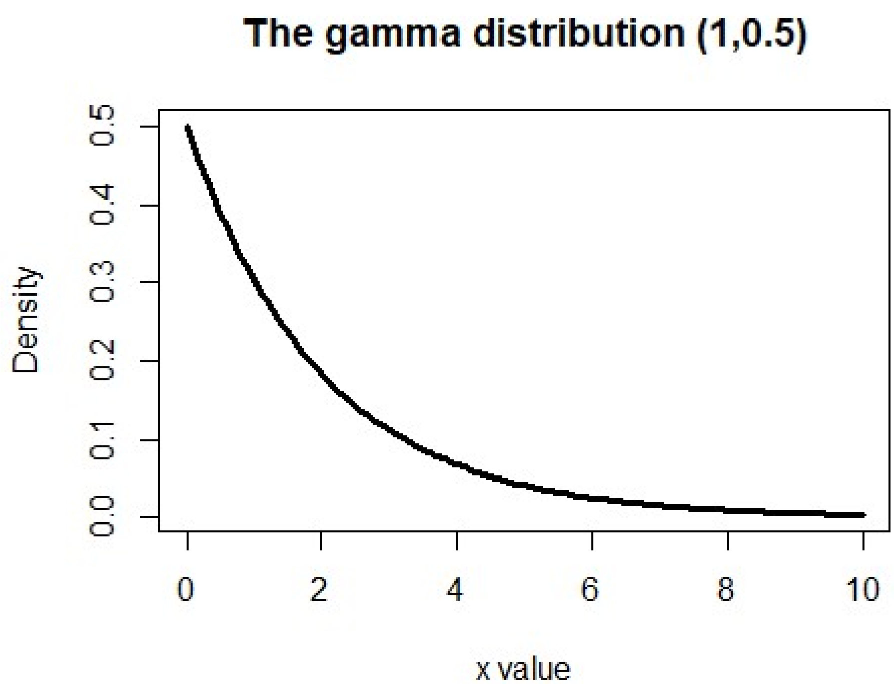
```
blavaan 0.5.4 ended normally after 1000 iterations

  Estimator                                      BAYES
  Optimization method                             MCMC
  Number of model parameters                        13

  Number of observations                           120

  Statistic                                 MargLogLik         PPP
  Value                                       -714.667       0.009

Parameter Estimates:
```

```
Latent Variables:
                   Estimate  Post.SD pi.lower pi.upper   Std.lv  Std.all
  Positive =~
    great              0.624    0.070    0.488    0.762    0.624    0.748
    cheerful           0.756    0.071    0.625    0.897    0.756    0.861
    happy              0.717    0.067    0.593    0.856    0.717    0.868
  Negative =~
    sad                0.561    0.070    0.426    0.703    0.561    0.733
    down               0.438    0.065    0.312    0.571    0.438    0.648
    unhappy            0.576    0.058    0.464    0.691    0.576    0.887
     Rhat     neff    Prior        Post.Med

    1.002 2554.022    normal(0,10)    0.623
    1.000 2128.213    normal(0,10)    0.753
    1.001 2108.183    normal(0,10)    0.715

    1.000 2688.437    normal(0,10)    0.561
    1.005 1619.912    normal(0,10)    0.437
    1.002 2111.591    normal(0,10)    0.575

Covariances:
                   Estimate  Post.SD pi.lower pi.upper   Std.lv  Std.all
  Positive ~~
    Negative          -0.309    0.105   -0.507   -0.098   -0.309   -0.309
     Rhat     neff    Prior        Post.Med

    1.000 3003.044       beta(1,1)   -0.313

Variances:
                   Estimate  Post.SD pi.lower pi.upper   Std.lv  Std.all
   .great              0.307    0.050    0.218    0.414    0.307    0.441
   .cheerful           0.200    0.051    0.106    0.303    0.200    0.259
   .happy              0.168    0.045    0.081    0.262    0.168    0.246
   .sad                0.270    0.051    0.177    0.376    0.270    0.462
   .down               0.265    0.046    0.183    0.360    0.265    0.580
   .unhappy            0.089    0.042    0.006    0.171    0.089    0.212
    Positive           1.000                               1.000    1.000
    Negative           1.000                               1.000    1.000
     Rhat     neff    Prior        Post.Med
    1.000 3067.617 gamma(1,.5)[sd]    0.304
    1.000 2007.101 gamma(1,.5)[sd]    0.198
    1.000 2544.036 gamma(1,.5)[sd]    0.166
    1.002 2385.310 gamma(1,.5)[sd]    0.267
    1.005 2056.642 gamma(1,.5)[sd]    0.262
    1.006 1194.727 gamma(1,.5)[sd]    0.090
                NA                       NA
                NA                       NA
```

```
R-Square:
                   Estimate
    great             0.559
    cheerful          0.741
    happy             0.754
    sad               0.538
    down              0.420
    unhappy           0.788
```

The prior for variance is:

```
# Gammma(1,0.5)

plot(seq(0,10,.1), dgamma(seq(0,10,.1),1,0.5), type="l", lty=1, lwd = 3,
xlab="x value",
     ylab="Density", main="The gamma distribution (1,0.5)")
```



And the model fit measures:

```
fitMeasures(modelfit.cfa1)
```

```
Warning:
6 (5.0%) p_waic estimates greater than 0.4. We recommend trying loo instead.

       npar       logl        ppp        bic        dic      p_dic       waic
     13.000   -664.192      0.009   1390.513   1354.203     12.909   1354.488
     p_waic    se_waic      looic      p_loo     se_loo margloglik
     12.746     39.509   1354.555     12.779     39.521   -714.667
```

In case we want to report the standardized posterior distribution of a latent variable model. All the chains are standardized and we can compute everything about them.

```
std_all <- standardizedposterior(modelfit.cfa1)
head(std_all)
```

```
     Positive=~great Positive=~cheerful Positive=~happy Negative=~sad
[1,]       0.7050249          0.9172129       0.8203739     0.7925537
[2,]       0.7995489          0.7642732       0.8749875     0.6943221
[3,]       0.7799854          0.9215920       0.8269719     0.7931555
[4,]       0.7028592          0.8855804       0.8601724     0.6795778
[5,]       0.6823058          0.8640067       0.8460113     0.6781016
[6,]       0.7302453          0.8568476       0.8313602     0.7619811
     Negative=~down Negative=~unhappy great~~great cheerful~~cheerful
[1,]      0.6371933         0.7770720    0.5029399          0.1587204
[2,]      0.6768404         0.9627449    0.3607216          0.4158864
[3,]      0.6046793         0.8997495    0.3916228          0.1506682
[4,]      0.7283661         0.9153657    0.5059890          0.2157473
[5,]      0.6584201         0.8917079    0.5344589          0.2534924
[6,]      0.6061246         0.8443196    0.4667418          0.2658121
     happy~~happy  sad~~sad down~~down unhappy~~unhappy Positive~~Positive
[1,]    0.3269867 0.3718586  0.5939847       0.39615917                  1
[2,]    0.2343969 0.5179168  0.5418870       0.07312219                  1
[3,]    0.3161174 0.3709044  0.6343630       0.19045093                  1
[4,]    0.2601034 0.5381740  0.4694828       0.16210557                  1
[5,]    0.2842649 0.5401782  0.5664829       0.20485701                  1
[6,]    0.3088402 0.4193848  0.6326130       0.28712444                  1
     Negative~~Negative Positive~~Negative
[1,]                  1         -0.3855060
[2,]                  1         -0.2404827
[3,]                  1         -0.3795258
[4,]                  1         -0.2938707
[5,]                  1         -0.2430206
[6,]                  1         -0.3771536
```

```
posterior_summary(std_all[,7:12])
```

```
                    Estimate  Est.Error       Q2.5     Q97.5
great~~great       0.4419335 0.07423190 0.30535638 0.5920537
cheerful~~cheerful 0.2609287 0.06925678 0.13429943 0.4077560
happy~~happy       0.2474291 0.06980573 0.11734840 0.3910257
sad~~sad           0.4625847 0.08827614 0.29685143 0.6439441
down~~down         0.5787107 0.09372508 0.39492958 0.7581193
unhappy~~unhappy   0.2132639 0.10194771 0.01218104 0.4157468
```

```
posterior_summary(1-std_all[,7:12]) ## R2
```

```
                    Estimate  Est.Error      Q2.5     Q97.5
great~~great       0.5580665 0.07423190 0.4079463 0.6946436
cheerful~~cheerful 0.7390713 0.06925678 0.5922440 0.8657006
happy~~happy       0.7525709 0.06980573 0.6089743 0.8826516
```

```
sad~~sad            0.5374153 0.08827614 0.3560559 0.7031486
down~~down          0.4212893 0.09372508 0.2418807 0.6050704
unhappy~~unhappy    0.7867361 0.10194771 0.5842532 0.9878190
```

3.3 Cross-loadings

We can define cross-loadings. For instance, we can assume that sadness/melancolia might be an indicator of positive feelings. Then, our second model is:

```
model.cfa2 <- 'Positive =~ great + cheerful + happy + sad
                Negative =~ sad + down + unhappy'

modelfit.cfa2 <- bcfa(model.cfa2, data=dat, std.lv=T, n.chains = 3,
                      burnin=5000, sample=1000, target = "stan")
```

```
Computing post-estimation metrics (including lvs if requested)...
```

```
summary(modelfit.cfa2 , standardized=T, rsquare=T, postmedian=TRUE)
```

```
blavaan 0.5.4 ended normally after 1000 iterations

  Estimator                                      BAYES
  Optimization method                             MCMC
  Number of model parameters                        14

  Number of observations                           120

  Statistic                                 MargLogLik         PPP
  Value                                       -713.640       0.102

Parameter Estimates:


Latent Variables:
                   Estimate  Post.SD pi.lower pi.upper   Std.lv  Std.all
  Positive =~
    great             0.631    0.069    0.503    0.769    0.631    0.755
    cheerful          0.757    0.072    0.620    0.903    0.757    0.860
    happy             0.716    0.067    0.585    0.852    0.716    0.865
    sad               0.293    0.103    0.112    0.511    0.293    0.381
  Negative =~
    sad               0.738    0.102    0.556    0.954    0.738    0.961
    down              0.475    0.062    0.358    0.600    0.475    0.702
    unhappy           0.505    0.060    0.392    0.621    0.505    0.780
     Rhat    Prior          Post.Med

    1.000    normal(0,10)    0.630
    1.000    normal(0,10)    0.753
    1.000    normal(0,10)    0.714
    1.001    normal(0,10)    0.286
```

```
    1.001    normal(0,10)    0.733
    1.002    normal(0,10)    0.472
    1.000    normal(0,10)    0.505

Covariances:
                   Estimate  Post.SD pi.lower pi.upper   Std.lv  Std.all
  Positive ~~
    Negative         -0.478    0.106   -0.669   -0.253   -0.478   -0.478
     Rhat    Prior         Post.Med

    1.001       beta(1,1)   -0.483

Variances:
                   Estimate  Post.SD pi.lower pi.upper   Std.lv  Std.all
   .great             0.300    0.049    0.215    0.406    0.300    0.430
   .cheerful          0.202    0.049    0.111    0.303    0.202    0.261
   .happy             0.172    0.041    0.098    0.257    0.172    0.251
   .sad               0.166    0.066    0.023    0.294    0.166    0.282
   .down              0.232    0.042    0.162    0.322    0.232    0.508
   .unhappy           0.164    0.038    0.091    0.246    0.164    0.392
    Positive          1.000                               1.000    1.000
    Negative          1.000                               1.000    1.000
     Rhat    Prior         Post.Med
    1.000 gamma(1,.5)[sd]    0.296
    0.999 gamma(1,.5)[sd]    0.200
    1.001 gamma(1,.5)[sd]    0.170
    1.003 gamma(1,.5)[sd]    0.168
    1.001 gamma(1,.5)[sd]    0.228
    1.000 gamma(1,.5)[sd]    0.164
                                NA
                                NA

R-Square:
                   Estimate
    great             0.570
    cheerful          0.739
    happy             0.749
    sad               0.718
    down              0.492
    unhappy           0.608
```

```
fitMeasures(modelfit.cfa2)
```

```
Warning:
8 (6.7%) p_waic estimates greater than 0.4. We recommend trying loo instead.

      npar       logl        ppp        bic        dic      p_dic       waic
    14.000   -658.191      0.102   1383.290   1344.393     14.005   1344.797
```

```
    p_waic    se_waic      looic      p_loo     se_loo margloglik
    13.865     39.081   1344.878     13.905     39.091   -713.640
```

Now, we can compare these two models. Apart from improved fit without compromising complexity, models must have theoretical ground:

```
blavCompare(modelfit.cfa1, modelfit.cfa2 )
```

```
Warning:
6 (5.0%) p_waic estimates greater than 0.4. We recommend trying loo instead.

Warning:
8 (6.7%) p_waic estimates greater than 0.4. We recommend trying loo instead.


WAIC estimates: 
 object1:  1354.488 
 object2:  1344.797 

 ELPD difference & SE: 
   -4.845    3.692 

LOO estimates: 
 object1:  1354.554 
 object2:  1344.877 

 ELPD difference & SE: 
   -4.838    3.694 

Laplace approximation to the log-Bayes factor
(experimental; positive values favor object1):   -1.027
```

We have the posteriors of each parameter:

```
mcs <- blavInspect(modelfit.cfa2, "mcmc")
mcs <- as.matrix(mcs)
head(mcs)
```

```
     Positive=~great Positive=~cheerful Positive=~happy Positive=~sad
[1,]       0.6651036          0.7960282       0.7913089     0.1323176
[2,]       0.5381409          0.6058160       0.7102524     0.3478869
[3,]       0.6940113          0.8022891       0.6430157     0.5836520
[4,]       0.7903476          0.8461303       0.7038235     0.5917355
[5,]       0.5122718          0.6452353       0.6152904     0.3910595
[6,]       0.7755557          0.7836699       0.7303341     0.4633415
     Negative=~sad Negative=~down Negative=~unhappy great~~great
[1,]     0.6228858      0.4093439         0.5703935    0.3033264
[2,]     0.7314256      0.5114637         0.4945096    0.3054523
[3,]     0.9153952      0.4618563         0.3666272    0.3662518
[4,]     0.9269297      0.4337478         0.4758977    0.2309271
[5,]     0.8025732      0.4357822         0.4290134    0.3649681
```

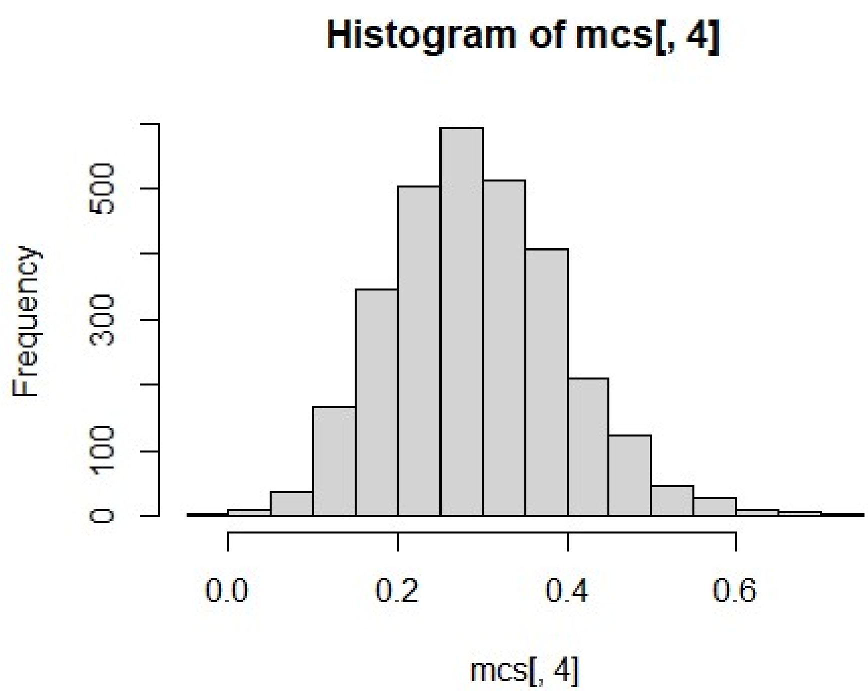
```
[6,]     0.9646861      0.4130075         0.4695733    0.2333594
     cheerful~~cheerful happy~~happy    sad~~sad down~~down unhappy~~unhappy
[1,]          0.1528991    0.2200524 0.182022625  0.2736850       0.09297277
[2,]          0.2292559    0.1645058 0.160956473  0.1964188       0.20547415
[3,]          0.2004209    0.1877266 0.053180564  0.2014021       0.20335322
[4,]          0.2363506    0.2426977 0.101928867  0.2047504       0.23766290
[5,]          0.1785642    0.1140468 0.009078291  0.3044407       0.24751816
[6,]          0.2340322    0.2400660 0.026858811  0.2125550       0.21365784
     Positive~~Negative
[1,]         -0.3981197
[2,]         -0.5656892
[3,]         -0.6702736
[4,]         -0.6792471
[5,]         -0.5383117
[6,]         -0.6437442
```

```
dim(mcs)
```

```
[1] 3000   14
```

And the histogram of the cross-loading is:

```
hist(mcs[,4])
```



The package psych can describe nicely these values:

```
describe(mcs)
```

```
                      vars    n  mean   sd median trimmed  mad   min   max range
Positive=~great          1 3000  0.63 0.07   0.63    0.63 0.07  0.41  0.93  0.51
Positive=~cheerful       2 3000  0.76 0.07   0.75    0.76 0.07  0.50  1.04  0.54
Positive=~happy          3 3000  0.72 0.07   0.71    0.72 0.07  0.51  0.95  0.43
Positive=~sad            4 3000  0.29 0.10   0.29    0.29 0.10 -0.02  0.73  0.75
Negative=~sad            5 3000  0.74 0.10   0.73    0.73 0.10  0.41  1.14  0.73
Negative=~down           6 3000  0.47 0.06   0.47    0.47 0.06  0.27  0.75  0.48
Negative=~unhappy        7 3000  0.50 0.06   0.50    0.50 0.06  0.33  0.76  0.43
great~~great             8 3000  0.30 0.05   0.30    0.30 0.05  0.17  0.50  0.33
cheerful~~cheerful       9 3000  0.20 0.05   0.20    0.20 0.05  0.04  0.41  0.37
happy~~happy            10 3000  0.17 0.04   0.17    0.17 0.04  0.05  0.37  0.32
sad~~sad                11 3000  0.17 0.07   0.17    0.17 0.06  0.00  0.40  0.40
down~~down              12 3000  0.23 0.04   0.23    0.23 0.04  0.12  0.46  0.34
unhappy~~unhappy        13 3000  0.16 0.04   0.16    0.16 0.04  0.02  0.30  0.28
Positive~~Negative      14 3000 -0.48 0.11  -0.48   -0.48 0.11 -0.75 -0.08  0.67
                       skew kurtosis se
Positive=~great        0.14    -0.05  0
Positive=~cheerful     0.18     0.08  0
Positive=~happy        0.12    -0.06  0
Positive=~sad          0.40     0.36  0
Negative=~sad          0.33     0.35  0
Negative=~down         0.15     0.09  0
Negative=~unhappy      0.15     0.11  0
great~~great           0.47     0.30  0
cheerful~~cheerful     0.26     0.31  0
happy~~happy           0.35     0.47  0
sad~~sad              -0.07     0.10  0
down~~down             0.56     0.75  0
unhappy~~unhappy       0.11     0.23  0
Positive~~Negative     0.33     0.05  0
```

And we can compute the probability of this cross-loading to be, for instance, higher than 0.1:

```
sum(mcs[,4] > .1)/nrow(mcs)
```

```
[1] 0.9843333
```

The function partable gives us the list of parameters in the model. In particular, it is useful to check the parameters that are fixed and the ones that are estimated.

```
partable(modelfit.cfa2)
```

```
   id      lhs op      rhs user block group free ustart exo label plabel
start
1   1 Positive =~    great    1     1     1    1     NA   0          .p1.
1.000
2   2 Positive =~ cheerful    1     1     1    2     NA   0          .p2.
1.000
3   3 Positive =~    happy    1     1     1    3     NA   0          .p3.
1.000
```

```
4   4 Positive =~      sad     1      1      1    4      NA    0          .p4.
1.000
5   5 Negative =~      sad     1      1      1    5      NA    0          .p5.
1.000
6   6 Negative =~     down     1      1      1    6      NA    0          .p6.
1.000
7   7 Negative =~  unhappy     1      1      1    7      NA    0          .p7.
1.000
8   8    great ~~    great     0      1      1    8      NA    0          .p8.
0.334
9   9 cheerful ~~ cheerful     0      1      1    9      NA    0          .p9.
0.368
10 10    happy ~~    happy     0      1      1   10      NA    0         .p10.
0.327
11 11      sad ~~      sad     0      1      1   11      NA    0         .p11.
0.280
12 12     down ~~     down     0      1      1   12      NA    0         .p12.
0.221
13 13  unhappy ~~  unhappy     0      1      1   13      NA    0         .p13.
0.202
14 14 Positive ~~ Positive     0      1      1    0       1    0         .p14.
1.000
15 15 Negative ~~ Negative     0      1      1    0       1    0         .p15.
1.000
16 16 Positive ~~ Negative     0      1      1   14      NA    0         .p16.
0.000
      est    se           prior stanpnum stansumnum  psrf      pxnames    mat
1   0.631 0.069     normal(0,10)        1          1 1.000   ly_sign[1] lambda
2   0.757 0.072     normal(0,10)        2          2 1.000   ly_sign[2] lambda
3   0.716 0.067     normal(0,10)        3          3 1.000   ly_sign[3] lambda
4   0.293 0.103     normal(0,10)        4          4 1.001   ly_sign[4] lambda
5   0.738 0.102     normal(0,10)        5          5 1.001   ly_sign[5] lambda
6   0.475 0.062     normal(0,10)        6          6 1.002   ly_sign[6] lambda
7   0.505 0.060     normal(0,10)        7          7 1.000   ly_sign[7] lambda
8   0.300 0.049 gamma(1,.5)[sd]        8          8 1.000 Theta_var[1]  theta
9   0.202 0.049 gamma(1,.5)[sd]        9          9 0.999 Theta_var[2]  theta
10  0.172 0.041 gamma(1,.5)[sd]       10         10 1.001 Theta_var[3]  theta
11  0.166 0.066 gamma(1,.5)[sd]       11         11 1.003 Theta_var[4]  theta
12  0.232 0.042 gamma(1,.5)[sd]       12         12 1.001 Theta_var[5]  theta
13  0.164 0.038 gamma(1,.5)[sd]       13         13 1.000 Theta_var[6]  theta
14  1.000 0.000                       NA         NA    NA         <NA>    psi
15  1.000 0.000                       NA         NA    NA         <NA>    psi
16 -0.478 0.106        beta(1,1)       14         14 1.001   Psi_cov[1]    psi
   row col logBF
1    1   1    NA
2    2   1    NA
3    3   1    NA
4    4   1    NA
5    4   2    NA
6    5   2    NA
```

```
7    6   2    NA
8    1   1    NA
9    2   2    NA
10   3   3    NA
11   4   4    NA
12   5   5    NA
13   6   6    NA
14   1   1    NA
15   2   2    NA
16   1   2    NA
```

Using the function hypothesis in package brms, we can compute PPP:

```
colnames(mcs) <-
c("Pg","Pc","Ph","Ps","Ns","Nd","Nu","gg","cc","hh","ss","dd","uu","PN")
hypothesis(mcs, "Ps > .1")
```

```
Hypothesis Tests for class :
     Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio Post.Prob
Star
1 (Ps)-(.1) > 0     0.19       0.1     0.04     0.37      62.83      0.98
*
---
'CI': 90%-CI for one-sided and 95%-CI for two-sided hypotheses.
'*': For one-sided hypotheses, the posterior probability exceeds 95%;
for two-sided hypotheses, the value tested against lies outside the 95%-CI.
Posterior probabilities of point hypotheses assume equal prior probabilities.
```

3.4 Allow covariance between residual variables

```
model.cfa3 <- 'Positive =~ great + cheerful + happy + sad
               Negative =~ sad + down + unhappy
               down ~~ unhappy'

modelfit.cfa3 <- bcfa(model.cfa3, data=dat, std.lv=T, n.chains = 3,
burnin=5000, sample=1000, target = "stan")
```

```
Warning: There were 5 divergent transitions after warmup. See
https://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup
to find out why this is a problem and how to eliminate them.

Warning: Examine the pairs() plot to diagnose sampling problems

Computing post-estimation metrics (including lvs if requested)...

Warning: blavaan WARNING: As specified, the theta covariance matrix is
neither diagonal nor unrestricted, so the actual prior might differ from the
stated prior. See
 https://arxiv.org/abs/2301.08667
```

```
summary(modelfit.cfa3, standardized=T,rsquare=T,postmedian=TRUE)
```

```
blavaan 0.5.4 ended normally after 1000 iterations

  Estimator                                      BAYES
  Optimization method                             MCMC
  Number of model parameters                        15

  Number of observations                           120

  Statistic                                 MargLogLik         PPP
  Value                                             NA       0.103

Parameter Estimates:


Latent Variables:
                   Estimate  Post.SD pi.lower pi.upper   Std.lv  Std.all
  Positive =~
    great             0.636    0.072    0.498    0.783    0.636    0.758
    cheerful          0.759    0.072    0.624    0.911    0.759    0.861
    happy             0.719    0.066    0.596    0.855    0.719    0.866
    sad               0.269    0.126    0.068    0.553    0.269    0.350
  Negative =~
    sad               0.708    0.142    0.482    1.011    0.708    0.923
    down              0.494    0.078    0.352    0.647    0.494    0.731
    unhappy           0.526    0.074    0.384    0.669    0.526    0.813
     Rhat    Prior        Post.Med

    1.000    normal(0,10)    0.635
    1.000    normal(0,10)    0.757
    1.000    normal(0,10)    0.717
    1.002    normal(0,10)    0.248

    1.003    normal(0,10)    0.686
    1.003    normal(0,10)    0.494
    1.005    normal(0,10)    0.529

Covariances:
                   Estimate  Post.SD pi.lower pi.upper   Std.lv  Std.all
 .down ~~
   .unhappy          -0.014    0.054   -0.096    0.086   -0.014   -0.081
  Positive ~~
    Negative         -0.459    0.111   -0.672   -0.239   -0.459   -0.459
     Rhat    Prior        Post.Med

    1.006       beta(1,1)   -0.019

    1.000       beta(1,1)   -0.461

Variances:
```

```
                   Estimate  Post.SD pi.lower pi.upper   Std.lv  Std.all
   .great             0.300    0.050    0.211    0.407    0.300    0.426
   .cheerful          0.200    0.049    0.111    0.300    0.200    0.258
   .happy             0.173    0.042    0.095    0.261    0.173    0.251
   .sad               0.190    0.110    0.001    0.364    0.190    0.323
   .down              0.213    0.062    0.100    0.338    0.213    0.466
   .unhappy           0.142    0.063    0.037    0.261    0.142    0.339
    Positive          1.000                               1.000    1.000
    Negative          1.000                               1.000    1.000
     Rhat    Prior         Post.Med
    1.000 gamma(1,.5)[sd]    0.297
    1.003 gamma(1,.5)[sd]    0.198
    1.000 gamma(1,.5)[sd]    0.172
    1.005 gamma(1,.5)[sd]    0.215
    1.003 gamma(1,.5)[sd]    0.211
    1.006 gamma(1,.5)[sd]    0.138
                                NA
                                NA

R-Square:
                   Estimate
    great             0.574
    cheerful          0.742
    happy             0.749
    sad               0.677
    down              0.534
    unhappy           0.661
```

```
fitMeasures(modelfit.cfa3)
```

```
Warning:
7 (5.8%) p_waic estimates greater than 0.4. We recommend trying loo instead.

      npar       logl        ppp        bic        dic      p_dic       waic
    15.000   -658.108      0.103   1387.904   1344.162     13.972   1344.313
    p_waic    se_waic      looic      p_loo     se_loo margloglik
    13.599     39.093   1344.438     13.662     39.112         NA
```

```
blavCompare(modelfit.cfa3, modelfit.cfa2)
```

```
Warning:
7 (5.8%) p_waic estimates greater than 0.4. We recommend trying loo instead.

Warning:
8 (6.7%) p_waic estimates greater than 0.4. We recommend trying loo instead.


WAIC estimates:
 object1:  1344.313
 object2:  1344.797
```

```
 ELPD difference & SE:
   -0.242    0.378

LOO estimates:
 object1:  1344.437
 object2:  1344.877

 ELPD difference & SE:
   -0.220    0.379

Laplace approximation to the log-Bayes factor
(experimental; positive values favor object1):       NA
```

```
blavCompare(modelfit.cfa3, modelfit.cfa1)
```

```
Warning:
7 (5.8%) p_waic estimates greater than 0.4. We recommend trying loo instead.
```

```
Warning:
6 (5.0%) p_waic estimates greater than 0.4. We recommend trying loo instead.
```

```
WAIC estimates:
 object1:  1344.313
 object2:  1354.488

 ELPD difference & SE:
   -5.088    3.487

LOO estimates:
 object1:  1344.437
 object2:  1354.554

 ELPD difference & SE:
   -5.059    3.489

Laplace approximation to the log-Bayes factor
(experimental; positive values favor object1):       NA
```

4. Model fit

We can use indicators of fit without null model:

```
ML_bs <- blavFitIndices(modelfit.cfa1)
```

```
Warning:
6 (5.0%) p_waic estimates greater than 0.4. We recommend trying loo instead.
```

```
ML_bs
```

```
Posterior mean (EAP) of devm-based fit indices:

      BRMSEA    BGammaHat adjBGammaHat          BMc
       0.150        0.941        0.850        0.911
```

```
summary(ML_bs, prob=.95,central.tendency = c("mean","median"))
```

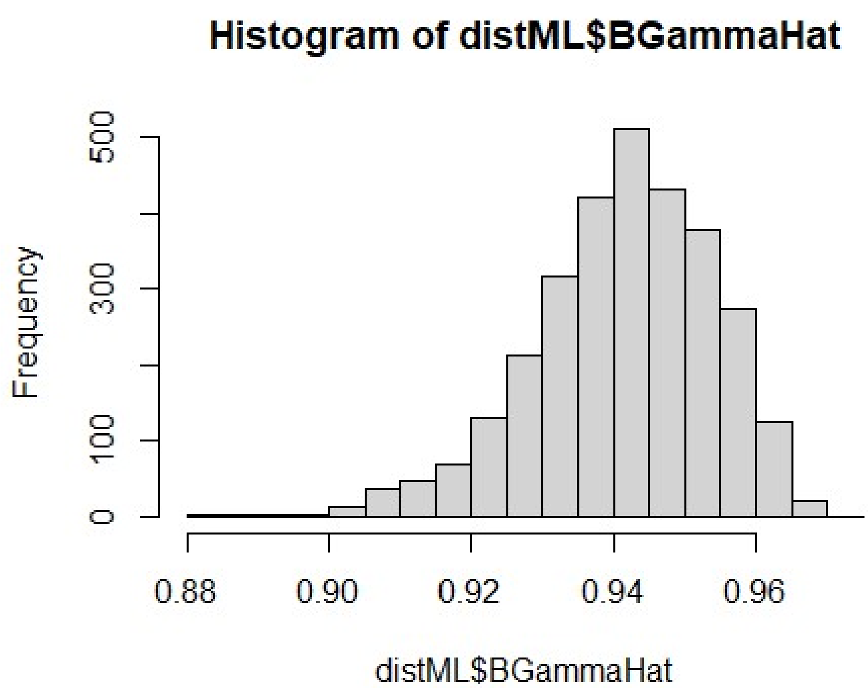
```
Posterior summary statistics and highest posterior density (HPD) 95% credible
intervals for devm-based fit indices:

               EAP Median    SD lower upper
BRMSEA       0.150  0.149 0.017 0.116 0.183
BGammaHat    0.941  0.943 0.013 0.917 0.965
adjBGammaHat 0.850  0.853 0.033 0.787 0.909
BMc          0.911  0.913 0.020 0.872 0.946
```

We can have access to the posterior distributions for further investigation:

```
distML <- data.frame(ML_bs@indices)
sum(distML$BGammaHat > .9)/nrow(distML)
```

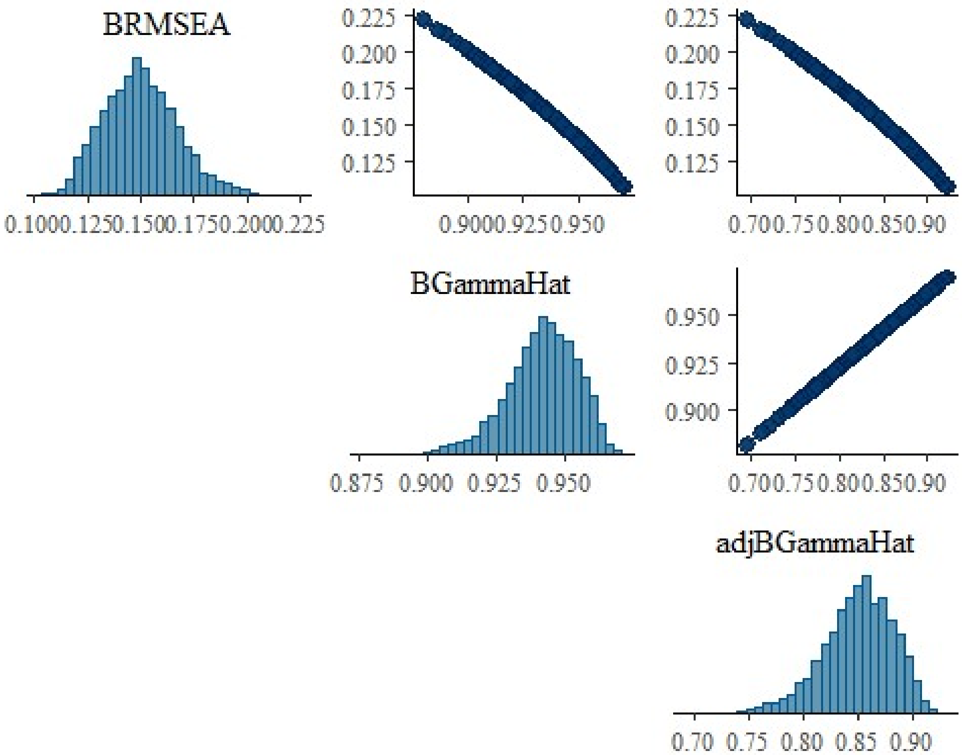
```
[1] 0.9966667
```

```
distML <- data.frame(ML_bs@indices)
hist(distML$BGammaHat)
```



```
mcmc_pairs(distML, pars = c("BRMSEA","BGammaHat","adjBGammaHat"), diag_fun =
"hist")
```

```
Warning: Only one chain in 'x'. This plot is more useful with multiple
chains.
```



4.1 Weakly informative priors

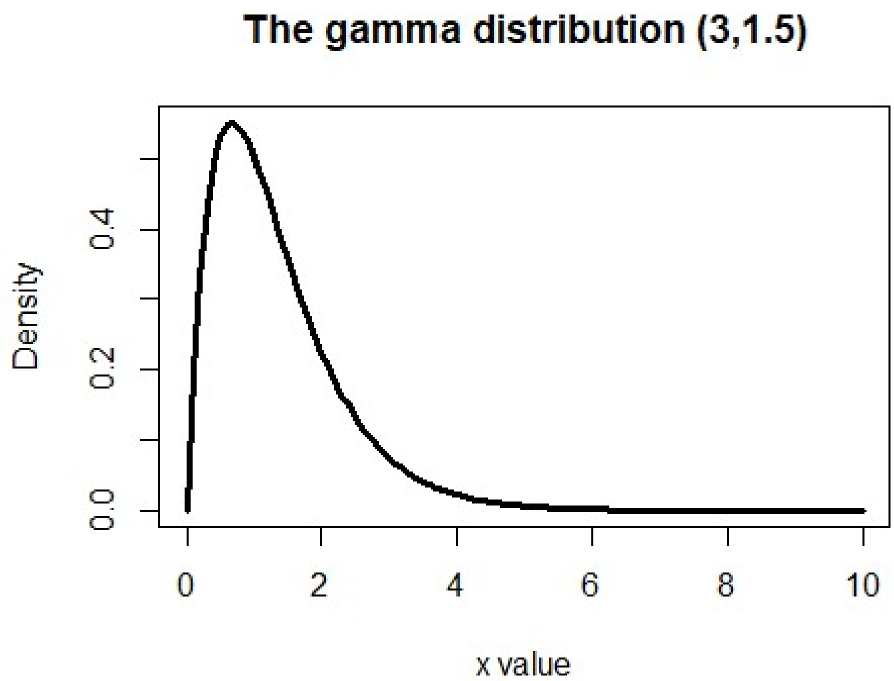
We can check the value of the priors using:

```
dpriors()
```

```
                 nu                alpha               lambda                 beta 
   "normal(0,32)"      "normal(0,10)"      "normal(0,10)"      "normal(0,10)" 
              theta                  psi                  rho                ibpsi 
"gamma(1,.5)[sd]" "gamma(1,.5)[sd]"        "beta(1,1)" "wishart(3,iden)" 
                tau 
  "normal(0,1.5)" 
```

We may want to change the prior of thetas from Gamma(1,0.5) to

```
# Gammma(1,0.5)

plot(seq(0,10,.1), dgamma(seq(0,10,.1),2,1.5), type="l", lty=1, lwd = 3,
xlab="x value",
     ylab="Density", main="The gamma distribution (3,1.5)")
```



Then we can change the priors:

```
modelfit.cfa1_dwp <- bcfa(model.cfa1, data=dat, std.lv=T, n.chains = 3,
burnin=10000, sample=5000, target="stan",
                    dp = dpriors(lambda="normal(0,100)", nu="normal(0,100)",
theta = "gamma(3,1.5)", target="stan"))
```

```
Computing post-estimation metrics (including lvs if requested)...
```

```
summary(modelfit.cfa1_dwp, standardized=T,rsquare=T,postmedian=TRUE)
```

```
blavaan 0.5.4 ended normally after 5000 iterations

  Estimator                                      BAYES
  Optimization method                             MCMC
  Number of model parameters                        13

  Number of observations                           120

  Statistic                                 MargLogLik         PPP
  Value                                       -743.249       0.004

Parameter Estimates:


Latent Variables:
                   Estimate  Post.SD pi.lower pi.upper   Std.lv  Std.all
```

```
  Positive =~
    great             0.629    0.071    0.497    0.772    0.629    0.749
    cheerful          0.747    0.070    0.616    0.891    0.747    0.846
    happy             0.705    0.066    0.582    0.842    0.705    0.845
  Negative =~
    sad               0.562    0.070    0.427    0.704    0.562    0.731
    down              0.466    0.064    0.344    0.595    0.466    0.681
    unhappy           0.524    0.057    0.415    0.639    0.524    0.794
     Rhat    Prior       Post.Med

    1.000 normal(0,100)    0.626
    1.000 normal(0,100)    0.745
    1.000 normal(0,100)    0.703

    1.000 normal(0,100)    0.562
    1.000 normal(0,100)    0.465
    1.000 normal(0,100)    0.523

Covariances:
                   Estimate  Post.SD pi.lower pi.upper   Std.lv  Std.all
  Positive ~~
    Negative         -0.349    0.107   -0.548   -0.132   -0.349   -0.349
     Rhat    Prior       Post.Med

    1.000     beta(1,1)   -0.353

Variances:
                   Estimate  Post.SD pi.lower pi.upper   Std.lv  Std.all
   .great             0.308    0.049    0.223    0.416    0.308    0.438
   .cheerful          0.222    0.043    0.147    0.315    0.222    0.284
   .happy             0.200    0.038    0.134    0.282    0.200    0.287
   .sad               0.276    0.051    0.187    0.390    0.276    0.466
   .down              0.252    0.043    0.174    0.344    0.252    0.537
   .unhappy           0.161    0.032    0.107    0.232    0.161    0.369
    Positive          1.000                               1.000    1.000
    Negative          1.000                               1.000    1.000
     Rhat    Prior       Post.Med
    1.000  gamma(3,1.5)    0.304
    1.000  gamma(3,1.5)    0.219
    1.000  gamma(3,1.5)    0.196
    1.000  gamma(3,1.5)    0.272
    1.000  gamma(3,1.5)    0.249
    1.000  gamma(3,1.5)    0.158
                              NA
                              NA

R-Square:
                   Estimate
    great             0.562
```

```
    cheerful          0.716
    happy             0.713
    sad               0.534
    down              0.463
    unhappy           0.631
```

```
fitMeasures(modelfit.cfa1_dwp)
```

```
Warning:
3 (2.5%) p_waic estimates greater than 0.4. We recommend trying loo instead.
```

```
      npar       logl        ppp        bic        dic      p_dic       waic
    13.000   -667.121      0.004   1396.370   1355.618     10.688   1355.505
    p_waic    se_waic      looic      p_loo     se_loo margloglik
    10.285     37.342   1355.543     10.304     37.350   -743.249
```

```
fits_st <- cbind(fitMeasures(modelfit.cfa1_dwp), fitMeasures(modelfit.cfa1))
```

```
Warning:
3 (2.5%) p_waic estimates greater than 0.4. We recommend trying loo instead.

Warning:
6 (5.0%) p_waic estimates greater than 0.4. We recommend trying loo instead.
```

```
round(fits_st,4)
```

```
                [,1]      [,2]
npar         13.0000   13.0000
logl       -667.1209 -664.1923
ppp           0.0043    0.0087
bic        1396.3705 1390.5132
dic        1355.6176 1354.2032
p_dic        10.6879   12.9093
waic       1355.5047 1354.4882
p_waic       10.2847   12.7455
se_waic      37.3417   39.5091
looic      1355.5429 1354.5549
p_loo        10.3039   12.7789
se_loo       37.3498   39.5213
margloglik -743.2494 -714.6666
```

5. Structural equation models (SEMs)

For SEM we need at least a regression connection between the variables (typically between the latent variables).

5.1 The PoliticalDemocracy data set

This is a famous dataset that is part of lavaan. The dataset contains various measures of political democracy and industrialization in 75 developing countries y1 - Expert ratings of the freedom of the press in 1960

y2 - The freedom of political opposition in 1960

y3 - The fairness of elections in 1960

y4 - The effectiveness of the elected legislature in 1960

y5 - Expert ratings of the freedom of the press in 1965

y6 - The freedom of political opposition in 1965

y7 - The fairness of elections in 1965

y8 - The effectiveness of the elected legislature in 1965

x1 - The gross national product (GNP) per capita in 1960

x2 - The inanimate energy consumption per capita in 1960

x3 - The percentage of the labor force in industry in 1960

```
head(PoliticalDemocracy)
```

```
     y1       y2       y3       y4       y5       y6       y7       y8
x1
1  2.50 0.000000 3.333333 0.000000 1.250000 0.000000 3.726360 3.333333
4.442651
2  1.25 0.000000 3.333333 0.000000 6.250000 1.100000 6.666666 0.736999
5.384495
3  7.50 8.800000 9.999998 9.199991 8.750000 8.094061 9.999998 8.211809
5.961005
4  8.90 8.800000 9.999998 9.199991 8.907948 8.127979 9.999998 4.615086
6.285998
5 10.00 3.333333 9.999998 6.666666 7.500000 3.333333 9.999998 6.666666
5.863631
6  7.50 3.333333 6.666666 6.666666 6.250000 1.100000 6.666666 0.368500
5.533389
        x2       x3
1 3.637586 2.557615
2 5.062595 3.568079
3 6.255750 5.224433
4 7.567863 6.267495
5 6.818924 4.573679
6 5.135798 3.892270
```

5.2 Model specification

```
model.sem <- '
  # measurement model
    ind60 =~ x1 + x2 + x3
    dem60 =~ y1 + y2 + y3 + y4
    dem65 =~ y5 + y6 + y7 + y8
```

```
  # regressions
    dem60 ~ ind60
    dem65 ~ ind60 + dem60
'
```

5.3 Model estimation

Now we can run the SEM model:

```
modelfit.sem <-  bcfa(model.sem, data=PoliticalDemocracy, std.lv=T,
                      n.chains = 4, burnin=10000,
                      sample=20000, target = "stan")
```

```
Computing post-estimation metrics (including lvs if requested)...
```

The estimates are:

```
summary(modelfit.sem, standardized=T,
        rsquare=T, neff=TRUE, postmedian=T)
```

```
blavaan 0.5.4 ended normally after 20000 iterations

  Estimator                                      BAYES
  Optimization method                             MCMC
  Number of model parameters                        25

  Number of observations                            75

  Statistic                                 MargLogLik         PPP
  Value                                      -1650.919       0.030

Parameter Estimates:


Latent Variables:
                   Estimate  Post.SD pi.lower pi.upper   Std.lv  Std.all
  ind60 =~
    x1                0.698    0.071    0.570    0.849    0.698    0.919
    x2                1.532    0.140    1.281    1.831    1.532    0.977
    x3                1.268    0.141    1.014    1.568    1.268    0.872
  dem60 =~
    y1                2.049    0.250    1.593    2.572    2.292    0.847
    y2                2.785    0.389    2.068    3.600    3.115    0.769
    y3                2.149    0.330    1.534    2.835    2.403    0.715
    y4                2.686    0.308    2.125    3.332    3.004    0.869
  dem65 =~
    y5                0.534    0.181    0.204    0.901    2.460    0.838
    y6                0.684    0.239    0.253    1.175    3.149    0.831
    y7                0.694    0.238    0.262    1.179    3.195    0.859
    y8                0.714    0.247    0.266    1.219    3.290    0.887
     Rhat     neff     Prior       Post.Med
```

```
    1.000 36866.232    normal(0,10)    0.693
    1.000 34892.575    normal(0,10)    1.523
    1.000 39957.579    normal(0,10)    1.260

    1.000 46061.880    normal(0,10)    2.037
    1.000 48321.218    normal(0,10)    2.768
    1.000 55190.188    normal(0,10)    2.135
    1.000 46312.465    normal(0,10)    2.669

    1.000 12399.517    normal(0,10)    0.529
    1.000 13233.993    normal(0,10)    0.675
    1.000 12872.342    normal(0,10)    0.687
    1.000 12538.400    normal(0,10)    0.706

Regressions:
                   Estimate  Post.SD pi.lower pi.upper   Std.lv  Std.all
  dem60 ~
    ind60             0.501    0.150    0.220    0.807    0.448    0.448
  dem65 ~
    ind60             0.685    0.467   -0.005    1.849    0.149    0.149
    dem60             3.708    1.767    1.748    8.688    0.900    0.900
     Rhat     neff      Prior        Post.Med

    1.000 59479.395    normal(0,10)    0.497

    1.000 16989.778    normal(0,10)    0.612
    1.000  7598.065    normal(0,10)    3.232

Variances:
                   Estimate  Post.SD pi.lower pi.upper   Std.lv  Std.all
   .x1                0.090    0.022    0.051    0.138    0.090    0.156
   .x2                0.114    0.077    0.001    0.283    0.114    0.046
   .x3                0.507    0.100    0.338    0.731    0.507    0.240
   .y1                2.062    0.458    1.306    3.083    2.062    0.282
   .y2                6.700    1.294    4.556    9.614    6.700    0.408
   .y3                5.520    1.022    3.835    7.829    5.520    0.489
   .y4                2.913    0.682    1.751    4.414    2.913    0.244
   .y5                2.567    0.517    1.716    3.737    2.567    0.298
   .y6                4.443    0.873    2.996    6.410    4.443    0.309
   .y7                3.624    0.725    2.422    5.253    3.624    0.262
   .y8                2.935    0.649    1.849    4.381    2.935    0.213
    ind60             1.000                               1.000    1.000
   .dem60             1.000                               0.799    0.799
   .dem65             1.000                               0.047    0.047
     Rhat     neff      Prior        Post.Med
    1.000 57707.639 gamma(1,.5)[sd]    0.089
    1.000 43824.665 gamma(1,.5)[sd]    0.107
    1.000 81787.152 gamma(1,.5)[sd]    0.497
```

```
    1.000 86209.046 gamma(1,.5)[sd]    2.015
    1.000 92567.704 gamma(1,.5)[sd]    6.564
    1.000 95166.334 gamma(1,.5)[sd]    5.409
    1.000 69852.780 gamma(1,.5)[sd]    2.849
    1.000 86824.682 gamma(1,.5)[sd]    2.512
    1.000 87347.573 gamma(1,.5)[sd]    4.354
    1.000 92988.233 gamma(1,.5)[sd]    3.547
    1.000 71824.997 gamma(1,.5)[sd]    2.870
                 NA                       NA
                 NA                       NA
                 NA                       NA

R-Square:
                   Estimate
    x1                0.844
    x2                0.954
    x3                0.760
    y1                0.718
    y2                0.592
    y3                0.511
    y4                0.756
    y5                0.702
    y6                0.691
    y7                0.738
    y8                0.787
    dem60             0.201
    dem65             0.953
```