Bifactor Models

Fernanda Andrade

The purpose of this exercise was to test the factor structure of a measure of habit development created for a study on behavior changes during the COVID-19 pandemic. The measure had 9 items and the response options were on a scale of 1-5.

Response scale: *1, not at all true of me*, to *5, very true of me*

Since the outbreak,

- newhabit01 = I have established a work or schoolwork routine
- newhabit02 = I have created a new schedule and have kept it somewhat consistently
- newhabit03 = I have restarted doing my usual tasks without realizing I'm doing them
- newhabit04 = I have established new regular routines (e.g., morning, bedtime routines)
- newhabit05 = I have found ways to accomplish the same activities I used to get done before the outbreak
- newhabit06 = I have found ways to include new tasks or obligations in my schedule
- newhabit07 = *I have created or assigned* a specific *space* to complete my work and school activities
- newhabit08 = *I have created or assigned* a specific *place* to exercise or engage in my usual physical activities
- newhabit09 = *I have assigned* new *times* to complete my activities (e.g., work, study)

One-Factor CFA

We first fit a one-factor CFA to examine whether this was a one-factor model with no correlated residual covariances. From taking a peek at the scale items, you can imagine that it is unlikely that residuals are uncorrelated, as many items share the same stem (e.g., I have created. . .)

We first specified the model so that all indicators were freely estimated. This means we had to fix the variance of the factor to 1 for identification;

```
modelcfa <- 'f1 =~ NA*newhabit01 + newhabit02 + newhabit03 + newhabit04 + newhabit05 +
                   newhabit06 + newhabit07 + newhabit08 + newhabit09

                  f1~~1*f1'
```

Then we fit the model with MLR as the estimator and FIML to account for missing data;

```
fitcfa <- cfa(modelcfa,
              data = habit,
              missing = "fiml",
              estimator = "mlr")
```

- The lavaan warning is telling that two cases had no data on any of the variables. These two were excluded from the analysis.

And requested the model summary with standardized estimates, fit indices, and r-square.

```
summary(fitcfa,
        standardized = TRUE,
        fit.measures = TRUE,
        rsquare = TRUE)
```

```
## lavaan 0.6-20.2265 ended normally after 29 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                        27
##
##                                                   Used       Total
##   Number of observations                           430         432
##   Number of missing patterns                        40
##
## Model Test User Model:
##                                                Standard      Scaled
##   Test Statistic                                175.549     123.819
##   Degrees of freedom                                27          27
##   P-value (Chi-square)                           0.000       0.000
##   Scaling correction factor                                  1.418
##     Yuan-Bentler correction (Mplus variant)
##
## Model Test Baseline Model:
##
##   Test statistic                              1554.087    1043.454
##   Degrees of freedom                                36          36
##   P-value                                        0.000       0.000
##   Scaling correction factor                                  1.489
##
## User Model versus Baseline Model:
##
##   Comparative Fit Index (CFI)                    0.902       0.904
##   Tucker-Lewis Index (TLI)                       0.870       0.872
##
##   Robust Comparative Fit Index (CFI)                         0.902
##   Robust Tucker-Lewis Index (TLI)                            0.869
##
## Loglikelihood and Information Criteria:
##
##   Loglikelihood user model (H0)              -5273.233   -5273.233
##   Scaling correction factor                                  1.091
##       for the MLR correction
##   Loglikelihood unrestricted model (H1)      -5185.458   -5185.458
##   Scaling correction factor                                  1.254
##       for the MLR correction
##
##   Akaike (AIC)                               10600.465   10600.465
##   Bayesian (BIC)                             10710.187   10710.187
##   Sample-size adjusted Bayesian (SABIC)      10624.505   10624.505
##
## Root Mean Square Error of Approximation:
```

```
##
##   RMSEA                                          0.113       0.091
##   90 Percent confidence interval - lower         0.097       0.078
##   90 Percent confidence interval - upper         0.129       0.105
##   P-value H_0: RMSEA <= 0.050                    0.000       0.000
##   P-value H_0: RMSEA >= 0.080                    1.000       0.918
##
##   Robust RMSEA                                                0.117
##   90 Percent confidence interval - lower                      0.097
##   90 Percent confidence interval - upper                      0.139
##   P-value H_0: Robust RMSEA <= 0.050                          0.000
##   P-value H_0: Robust RMSEA >= 0.080                          0.998
##
## Standardized Root Mean Square Residual:
##
##   SRMR                                           0.052       0.052
##
## Parameter Estimates:
##
##   Standard errors                             Sandwich
##   Information bread                           Observed
##   Observed information based on                Hessian
##
## Latent Variables:
##                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##   f1 =~
##     newhabit01        0.876    0.061   14.378    0.000    0.876    0.712
##     newhabit02        1.025    0.044   23.078    0.000    1.025    0.795
##     newhabit03        0.749    0.060   12.473    0.000    0.749    0.617
##     newhabit04        0.877    0.054   16.216    0.000    0.877    0.679
##     newhabit05        0.822    0.052   15.773    0.000    0.822    0.696
##     newhabit06        0.732    0.057   12.874    0.000    0.732    0.658
##     newhabit07        0.832    0.063   13.301    0.000    0.832    0.652
##     newhabit08        0.839    0.068   12.328    0.000    0.839    0.603
##     newhabit09        0.873    0.056   15.491    0.000    0.873    0.677
##
## Intercepts:
##                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##    .newhabit01        3.402    0.062   55.255    0.000    3.402    2.766
##    .newhabit02        3.227    0.063   51.283    0.000    3.227    2.504
##    .newhabit03        2.908    0.059   49.072    0.000    2.908    2.393
##    .newhabit04        3.083    0.063   49.063    0.000    3.083    2.387
##    .newhabit05        3.264    0.057   56.883    0.000    3.264    2.765
##    .newhabit06        3.532    0.055   64.570    0.000    3.532    3.174
##    .newhabit07        3.517    0.065   54.300    0.000    3.517    2.757
##    .newhabit08        3.277    0.070   46.760    0.000    3.277    2.353
##    .newhabit09        3.208    0.064   50.337    0.000    3.208    2.489
##
## Variances:
##                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##     f1                1.000                              1.000    1.000
##    .newhabit01        0.747    0.087    8.552    0.000    0.747    0.493
##    .newhabit02        0.611    0.066    9.313    0.000    0.611    0.368
##    .newhabit03        0.915    0.091   10.063    0.000    0.915    0.620
```

```
##    .newhabit04        0.899    0.088   10.176    0.000    0.899    0.539
##    .newhabit05        0.718    0.066   10.943    0.000    0.718    0.515
##    .newhabit06        0.702    0.071    9.842    0.000    0.702    0.567
##    .newhabit07        0.935    0.090   10.389    0.000    0.935    0.575
##    .newhabit08        1.235    0.108   11.470    0.000    1.235    0.637
##    .newhabit09        0.899    0.086   10.488    0.000    0.899    0.541
##
## R-Square:
##                    Estimate
##     newhabit01        0.507
##     newhabit02        0.632
##     newhabit03        0.380
##     newhabit04        0.461
##     newhabit05        0.485
##     newhabit06        0.433
##     newhabit07        0.425
##     newhabit08        0.363
##     newhabit09        0.459
```

We can also request the residual covariance matrix to see the residual covariances.

```
lavInspect(fitcfa, what = "resid") # Look at the residual covariances (estimated)
```

```
## $cov
##            nwhb01 nwhb02 nwhb03 nwhb04 nwhb05 nwhb06 nwhb07 nwhb08 nwhb09
## newhabit01  0.003
## newhabit02  0.141 -0.006
## newhabit03  0.090  0.041 -0.001
## newhabit04 -0.099  0.124  0.039  0.002
## newhabit05  0.053 -0.051  0.023 -0.055 -0.001
## newhabit06 -0.090 -0.069 -0.064  0.032  0.161  0.001
## newhabit07 -0.035 -0.112 -0.092 -0.074 -0.031  0.045 -0.003
## newhabit08 -0.080 -0.109 -0.003 -0.119 -0.032 -0.010  0.235  0.006
## newhabit09 -0.118 -0.067 -0.087  0.020 -0.056 -0.014  0.179  0.319 -0.001
##
## $mean
## newhabit01 newhabit02 newhabit03 newhabit04 newhabit05 newhabit06 newhabit07 newhabit08
##      0.001      0.002      0.001     -0.005      0.002      0.000      0.003     -0.003
## newhabit09
##      0.004
```

Fit isn't good and you can see from the residual covariance matrix that items that share a question stem (or the same wording) are more strongly covarying (e.g., newhabit8 and newhabit9).

```
modindices(fitcfa, sort = TRUE)
```

```
##           lhs op        rhs     mi    epc sepc.lv sepc.all sepc.nox
## 65 newhabit08 ~~ newhabit09 42.933  0.401   0.401    0.381    0.381
## 30 newhabit01 ~~ newhabit02 26.302  0.231   0.231    0.342    0.342
## 56 newhabit05 ~~ newhabit06 23.547  0.197   0.197    0.278    0.278
## 63 newhabit07 ~~ newhabit08 19.530  0.280   0.280    0.261    0.261
## 64 newhabit07 ~~ newhabit09 19.512  0.238   0.238    0.259    0.259
```

```
## 39 newhabit02 ~~ newhabit04 17.799  0.198   0.198    0.267    0.267
## 42 newhabit02 ~~ newhabit07 10.436 -0.157  -0.157   -0.208   -0.208
## 37 newhabit01 ~~ newhabit09  9.779 -0.155  -0.155   -0.189   -0.189
## 43 newhabit02 ~~ newhabit08  8.815 -0.163  -0.163   -0.187   -0.187
## 54 newhabit04 ~~ newhabit08  6.945 -0.160  -0.160   -0.152   -0.152
## 34 newhabit01 ~~ newhabit06  6.469 -0.111  -0.111   -0.153   -0.153
## 31 newhabit01 ~~ newhabit03  5.247  0.111   0.111    0.134    0.134
## 32 newhabit01 ~~ newhabit04  5.203 -0.112  -0.112   -0.137   -0.137
## 48 newhabit03 ~~ newhabit07  4.758 -0.116  -0.116   -0.125   -0.125
## 50 newhabit03 ~~ newhabit09  4.638 -0.111  -0.111   -0.122   -0.122
## 44 newhabit02 ~~ newhabit09  4.465 -0.100  -0.100   -0.136   -0.136
## 36 newhabit01 ~~ newhabit08  4.097 -0.118  -0.118   -0.122   -0.122
## 47 newhabit03 ~~ newhabit06  3.540 -0.084  -0.084   -0.104   -0.104
## 41 newhabit02 ~~ newhabit06  2.981 -0.071  -0.071   -0.109   -0.109
## 51 newhabit04 ~~ newhabit05  2.906 -0.079  -0.079   -0.098   -0.098
## 40 newhabit02 ~~ newhabit05  2.795 -0.071  -0.071   -0.107   -0.107
## 33 newhabit01 ~~ newhabit05  2.748  0.074   0.074    0.101    0.101
## 59 newhabit05 ~~ newhabit09  2.555 -0.075  -0.075   -0.094   -0.094
## 53 newhabit04 ~~ newhabit07  2.453 -0.084  -0.084   -0.092   -0.092
## 38 newhabit02 ~~ newhabit03  2.116  0.066   0.066    0.089    0.089
## 60 newhabit06 ~~ newhabit07  1.699  0.062   0.062    0.076    0.076
## 58 newhabit05 ~~ newhabit08  1.491 -0.067  -0.067   -0.071   -0.071
## 45 newhabit03 ~~ newhabit04  1.296  0.057   0.057    0.063    0.063
## 35 newhabit01 ~~ newhabit07  0.844 -0.046  -0.046   -0.056   -0.056
## 52 newhabit04 ~~ newhabit06  0.791  0.040   0.040    0.051    0.051
## 46 newhabit03 ~~ newhabit05  0.480  0.031   0.031    0.039    0.039
## 57 newhabit05 ~~ newhabit07  0.358 -0.029  -0.029   -0.035   -0.035
## 55 newhabit04 ~~ newhabit09  0.220  0.024   0.024    0.027    0.027
## 62 newhabit06 ~~ newhabit09  0.136 -0.017  -0.017   -0.021   -0.021
## 61 newhabit06 ~~ newhabit08  0.118 -0.018  -0.018   -0.020   -0.020
## 49 newhabit03 ~~ newhabit08  0.113 -0.020  -0.020   -0.019   -0.019
```

```
# The sort argument allows us to sort the indices from largest to smallest
```

If we look at the modification indices, these covariances are indeed associated with the highest MIs, which indicates that freeing these residual covariances would improve model fit.

Scale unidimensionality using omega

We can also request the omega coefficient. This function called reliability() is from the *semTools* package. It provides several measures of reliability, including alpha, and three types of omega. The first and second omega estimates are calculated so that the denominator equals the model-implied variance of the total scores. The third omega is calculated so that the denominator equals the observed, sample variance of scores. To the extent that the model is a good fitting one, the second and third estimates of omega should be very similar (because the model-implied and observed matrices should be similar).

Omega is interpreted as the proportion of total-score variance that is due to - or explained by - a single, or general, factor. In other words, it tells whether an overall score is worth interpreting despite any multidimensionality of the construct. As noted in Reise et al. (Chapter 18), omega is influenced by the number of items: more items, higher omega because scores better reflect the general construct.

Omega is high and equals .88.

```
reliability(fitcfa) # Omega
```

```
##               f1
## alpha  0.8828567
## omega  0.8835894
## omega2 0.8835894
## omega3 0.8826630
## avevar 0.4597160
```

Two-Factor EFA

To explore whether variables with similar stems cluster together, we can run a two-factor EFA.

We specify the CFA model as we did previously in the class:

```
# these are exploratory blocks (you can give them any name), you'll need to specify them
efa_model <- 'efa("block1")*F1 =~ newhabit01 + newhabit02 + newhabit03 + newhabit04 +
                                  newhabit05 + newhabit06 + newhabit07 + newhabit08 +
                                  newhabit09
              efa("block1")*F2 =~ newhabit01 + newhabit02 + newhabit03 + newhabit04 +
                                  newhabit05 + newhabit06 + newhabit07 + newhabit08 +
                                  newhabit09'
```

We fit it.

```
# Estimate the Model
efa_f1 <-
  sem(model = efa_model,
      data = habit,
      rotation = "oblimin",
      estimator = "MLR"
  )
```

And request the output.

```
summary(efa_f1,
        fit.measures = TRUE,
        standardized = TRUE,
        rsquare = TRUE)
```

```
## lavaan 0.6-20.2265 ended normally after 1 iteration
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                        28
##   Row rank of the constraints matrix                 2
##
##   Rotation method                       OBLIMIN OBLIQUE
##   Oblimin gamma                                      0
##   Rotation algorithm (rstarts)                GPA (30)
##   Standardized metric                             TRUE
```

```
##   Row weights                                       None
##
##                                                    Used       Total
##   Number of observations                            317         432
##
## Model Test User Model:
##                                                 Standard      Scaled
##   Test Statistic                                  53.735      41.744
##   Degrees of freedom                                  19          19
##   P-value (Chi-square)                             0.000       0.002
##   Scaling correction factor                                    1.287
##     Yuan-Bentler correction (Mplus variant)
##
## Model Test Baseline Model:
##
##   Test statistic                                1391.843     951.686
##   Degrees of freedom                                  36          36
##   P-value                                          0.000       0.000
##   Scaling correction factor                                    1.463
##
## User Model versus Baseline Model:
##
##   Comparative Fit Index (CFI)                      0.974       0.975
##   Tucker-Lewis Index (TLI)                         0.951       0.953
##
##   Robust Comparative Fit Index (CFI)                           0.978
##   Robust Tucker-Lewis Index (TLI)                              0.959
##
## Loglikelihood and Information Criteria:
##
##   Loglikelihood user model (H0)                -3987.159   -3987.159
##   Scaling correction factor                                    1.285
##       for the MLR correction
##   Loglikelihood unrestricted model (H1)        -3960.291   -3960.291
##   Scaling correction factor                                    1.286
##       for the MLR correction
##
##   Akaike (AIC)                                  8026.318    8026.318
##   Bayesian (BIC)                                8124.049    8124.049
##   Sample-size adjusted Bayesian (SABIC)         8041.583    8041.583
##
## Root Mean Square Error of Approximation:
##
##   RMSEA                                            0.076       0.061
##   90 Percent confidence interval - lower           0.052       0.039
##   90 Percent confidence interval - upper           0.100       0.084
##   P-value H_0: RMSEA <= 0.050                      0.036       0.184
##   P-value H_0: RMSEA >= 0.080                      0.416       0.089
##
##   Robust RMSEA                                                 0.070
##   90 Percent confidence interval - lower                       0.041
##   90 Percent confidence interval - upper                       0.099
##   P-value H_0: Robust RMSEA <= 0.050                           0.120
##   P-value H_0: Robust RMSEA >= 0.080                           0.300
```

```
##
## Standardized Root Mean Square Residual:
##
##   SRMR                                           0.027       0.027
##
## Parameter Estimates:
##
##   Standard errors                             Sandwich
##   Information bread                           Observed
##   Observed information based on                Hessian
##
## Latent Variables:
##                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##   F1 =~ block1
##     newhabit01        1.023    0.087   11.765    0.000    1.023    0.837
##     newhabit02        1.122    0.083   13.557    0.000    1.122    0.862
##     newhabit03        0.868    0.097    8.910    0.000    0.868    0.718
##     newhabit04        0.753    0.128    5.890    0.000    0.753    0.603
##     newhabit05        0.654    0.124    5.254    0.000    0.654    0.565
##     newhabit06        0.399    0.151    2.649    0.008    0.399    0.364
##     newhabit07        0.121    0.130    0.927    0.354    0.121    0.095
##     newhabit08       -0.003    0.094   -0.030    0.976   -0.003   -0.002
##     newhabit09       -0.029    0.070   -0.409    0.683   -0.029   -0.023
##   F2 =~ block1
##     newhabit01       -0.052    0.077   -0.678    0.498   -0.052   -0.043
##     newhabit02       -0.010    0.076   -0.131    0.896   -0.010   -0.008
##     newhabit03       -0.086    0.096   -0.899    0.369   -0.086   -0.071
##     newhabit04        0.186    0.123    1.513    0.130    0.186    0.149
##     newhabit05        0.263    0.123    2.128    0.033    0.263    0.227
##     newhabit06        0.364    0.159    2.292    0.022    0.364    0.332
##     newhabit07        0.824    0.139    5.931    0.000    0.824    0.645
##     newhabit08        1.028    0.107    9.596    0.000    1.028    0.742
##     newhabit09        1.042    0.100   10.413    0.000    1.042    0.824
##
## Covariances:
##                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##   F1 ~~
##     F2                0.713    0.045   15.993    0.000    0.713    0.713
##
## Variances:
##                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##    .newhabit01        0.520    0.089    5.851    0.000    0.520    0.349
##    .newhabit02        0.451    0.071    6.376    0.000    0.451    0.266
##    .newhabit03        0.806    0.102    7.870    0.000    0.806    0.552
##    .newhabit04        0.756    0.098    7.731    0.000    0.756    0.486
##    .newhabit05        0.599    0.059   10.102    0.000    0.599    0.447
##    .newhabit06        0.703    0.081    8.694    0.000    0.703    0.585
##    .newhabit07        0.794    0.097    8.173    0.000    0.794    0.487
##    .newhabit08        0.869    0.113    7.716    0.000    0.869    0.452
##    .newhabit09        0.554    0.125    4.433    0.000    0.554    0.347
##     F1                1.000                              1.000    1.000
##     F2                1.000                              1.000    1.000
##
## R-Square:
```

```
##                    Estimate
##     newhabit01        0.651
##     newhabit02        0.734
##     newhabit03        0.448
##     newhabit04        0.514
##     newhabit05        0.553
##     newhabit06        0.415
##     newhabit07        0.513
##     newhabit08        0.548
##     newhabit09        0.653
```

The pattern of loadings is consistent with question wording: items 7-9 are loading on one factor and items 1-6 on another.

If we check the items again, it appears that items 1-6 are assessing structuring one's schedules or routines, and items 7-9 are assessing whether someone has assigned new times and spaces for their activities. Both of these (routines, context stability) are features of habits.

Methods factor CFA

To deal with these three items that seem to share some method variance, potentially due to the wording, we could add a method factor to the model. To make sure your model is empirically identified and conceptually sensible, the correlation between the method factor (f2) and the conceptual factor (f1) must be 0 - otherwise, we assume they share something in common, but that shared variance is already being captured by f1.

```
methodfactor <- 'f1 =~ NA*newhabit01 + newhabit02 + newhabit03 + newhabit04 + newhabit05 +
                    newhabit06 + newhabit07 + newhabit08 + newhabit09
                f2 =~ NA*newhabit07 + newhabit08 + newhabit09

                    f1~~1*f1
                    f2~~1*f2
                    f1~~0*f2
'
```

```
methodfactorfit <- sem(methodfactor,
               data = habit,
               missing = "fiml",
               estimator = "mlr")
```

```
summary(methodfactorfit,
        standardized = TRUE,
        fit.measures = TRUE,
        rsquare = TRUE)
```

```
## lavaan 0.6-20.2265 ended normally after 37 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                        30
##
##                                                   Used       Total
##   Number of observations                           430         432
```

```
##   Number of missing patterns                        40
##
## Model Test User Model:
##                                              Standard      Scaled
##   Test Statistic                               90.250      64.939
##   Degrees of freedom                               24          24
##   P-value (Chi-square)                          0.000       0.000
##   Scaling correction factor                                 1.390
##     Yuan-Bentler correction (Mplus variant)
##
## Model Test Baseline Model:
##
##   Test statistic                             1554.087    1043.454
##   Degrees of freedom                               36          36
##   P-value                                       0.000       0.000
##   Scaling correction factor                                 1.489
##
## User Model versus Baseline Model:
##
##   Comparative Fit Index (CFI)                   0.956       0.959
##   Tucker-Lewis Index (TLI)                      0.935       0.939
##
##   Robust Comparative Fit Index (CFI)                        0.958
##   Robust Tucker-Lewis Index (TLI)                           0.937
##
## Loglikelihood and Information Criteria:
##
##   Loglikelihood user model (H0)              -5230.583   -5230.583
##   Scaling correction factor                                 1.146
##       for the MLR correction
##   Loglikelihood unrestricted model (H1)      -5185.458   -5185.458
##   Scaling correction factor                                 1.254
##       for the MLR correction
##
##   Akaike (AIC)                               10521.166   10521.166
##   Bayesian (BIC)                             10643.079   10643.079
##   Sample-size adjusted Bayesian (SABIC)      10547.877   10547.877
##
## Root Mean Square Error of Approximation:
##
##   RMSEA                                         0.080       0.063
##   90 Percent confidence interval - lower        0.063       0.048
##   90 Percent confidence interval - upper        0.098       0.079
##   P-value H_0: RMSEA <= 0.050                   0.003       0.081
##   P-value H_0: RMSEA >= 0.080                   0.526       0.038
##
##   Robust RMSEA                                              0.081
##   90 Percent confidence interval - lower                    0.058
##   90 Percent confidence interval - upper                    0.105
##   P-value H_0: Robust RMSEA <= 0.050                        0.015
##   P-value H_0: Robust RMSEA >= 0.080                        0.566
##
## Standardized Root Mean Square Residual:
##
```

```
##   SRMR                                           0.035       0.035
##
## Parameter Estimates:
##
##   Standard errors                             Sandwich
##   Information bread                           Observed
##   Observed information based on                Hessian
##
## Latent Variables:
##                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##   f1 =~
##     newhabit01        0.900    0.059   15.217    0.000    0.900    0.731
##     newhabit02        1.058    0.043   24.824    0.000    1.058    0.821
##     newhabit03        0.763    0.059   12.832    0.000    0.763    0.628
##     newhabit04        0.890    0.055   16.187    0.000    0.890    0.689
##     newhabit05        0.826    0.053   15.507    0.000    0.826    0.700
##     newhabit06        0.727    0.058   12.477    0.000    0.727    0.653
##     newhabit07        0.755    0.066   11.505    0.000    0.755    0.593
##     newhabit08        0.749    0.068   11.052    0.000    0.749    0.537
##     newhabit09        0.794    0.058   13.762    0.000    0.794    0.616
##   f2 =~
##     newhabit07        0.494    0.098    5.061    0.000    0.494    0.387
##     newhabit08        0.734    0.103    7.099    0.000    0.734    0.527
##     newhabit09        0.625    0.103    6.076    0.000    0.625    0.485
##
## Covariances:
##                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##   f1 ~~
##     f2                0.000                               0.000    0.000
##
## Intercepts:
##                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##    .newhabit01        3.404    0.062   55.345    0.000    3.404    2.767
##    .newhabit02        3.228    0.063   51.324    0.000    3.228    2.505
##    .newhabit03        2.907    0.059   49.060    0.000    2.907    2.393
##    .newhabit04        3.083    0.063   49.072    0.000    3.083    2.387
##    .newhabit05        3.263    0.057   56.855    0.000    3.263    2.763
##    .newhabit06        3.531    0.055   64.502    0.000    3.531    3.173
##    .newhabit07        3.518    0.064   54.623    0.000    3.518    2.761
##    .newhabit08        3.275    0.070   46.729    0.000    3.275    2.349
##    .newhabit09        3.214    0.064   50.556    0.000    3.214    2.495
##
## Variances:
##                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##     f1                1.000                               1.000    1.000
##     f2                1.000                               1.000    1.000
##    .newhabit01        0.704    0.086    8.204    0.000    0.704    0.465
##    .newhabit02        0.541    0.062    8.739    0.000    0.541    0.326
##    .newhabit03        0.894    0.091    9.854    0.000    0.894    0.606
##    .newhabit04        0.876    0.090    9.701    0.000    0.876    0.525
##    .newhabit05        0.712    0.067   10.576    0.000    0.712    0.511
##    .newhabit06        0.711    0.073    9.785    0.000    0.711    0.574
##    .newhabit07        0.809    0.093    8.746    0.000    0.809    0.498
##    .newhabit08        0.845    0.143    5.921    0.000    0.845    0.434
```

```
##    .newhabit09        0.639    0.128    4.991    0.000    0.639    0.385
##
## R-Square:
##                    Estimate
##     newhabit01        0.535
##     newhabit02        0.674
##     newhabit03        0.394
##     newhabit04        0.475
##     newhabit05        0.489
##     newhabit06        0.426
##     newhabit07        0.502
##     newhabit08        0.566
##     newhabit09        0.615
```

```
reliability(methodfactorfit)
```

```
##               f1        f2
## alpha  0.8828567 0.7874316
## omega  0.8921357 0.5995169
## omega2 0.8456186 0.3118791
## omega3 0.8451412 0.3118590
## avevar        NA        NA
```

Two-Correlated Factors CFA

Alternatively, we could estimate a two correlated-factors CFA, where items 07, 08, and 09 load onto a second factor. This would be consistent with most measurement models with two correlated subscales.

However, it's important to rememeber that this process masks the overall construct hierarchy (e.g., a single construct with two subscales that capture meaningful variance specific to the subscale) as well as potentially meaningful underlying patterns of relationships between indicators. These misspecifications can lead to inflated correlations between the two factors.

```
twofactor <- 'f1 =~ NA*newhabit01 + newhabit02 + newhabit03 + newhabit04 + newhabit05 +
                    newhabit06
              f2 =~ NA*newhabit07 + newhabit08 + newhabit09

                    f1~~1*f1
                    f2~~1*f2
'
```

```
twofactorfit <- cfa(twofactor,
               data = habit,
               missing = "fiml",
               estimator = "mlr")
```

```
summary(twofactorfit,
        standardized = TRUE,
        fit.measures = TRUE,
        rsquare = TRUE)
```

```
## lavaan 0.6-20.2265 ended normally after 34 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                        28
##
##                                                   Used       Total
##   Number of observations                           430         432
##   Number of missing patterns                        40
##
## Model Test User Model:
##                                               Standard      Scaled
##   Test Statistic                                93.640      67.429
##   Degrees of freedom                                26          26
##   P-value (Chi-square)                           0.000       0.000
##   Scaling correction factor                                  1.389
##     Yuan-Bentler correction (Mplus variant)
##
## Model Test Baseline Model:
##
##   Test statistic                              1554.087    1043.454
##   Degrees of freedom                                36          36
##   P-value                                        0.000       0.000
##   Scaling correction factor                                  1.489
##
## User Model versus Baseline Model:
##
##   Comparative Fit Index (CFI)                    0.955       0.959
##   Tucker-Lewis Index (TLI)                       0.938       0.943
##
##   Robust Comparative Fit Index (CFI)                         0.958
##   Robust Tucker-Lewis Index (TLI)                            0.941
##
## Loglikelihood and Information Criteria:
##
##   Loglikelihood user model (H0)              -5232.278   -5232.278
##   Scaling correction factor                                  1.129
##       for the MLR correction
##   Loglikelihood unrestricted model (H1)      -5185.458   -5185.458
##   Scaling correction factor                                  1.254
##       for the MLR correction
##
##   Akaike (AIC)                               10520.556   10520.556
##   Bayesian (BIC)                             10634.342   10634.342
##   Sample-size adjusted Bayesian (SABIC)      10545.486   10545.486
##
## Root Mean Square Error of Approximation:
##
##   RMSEA                                          0.078       0.061
##   90 Percent confidence interval - lower         0.061       0.046
##   90 Percent confidence interval - upper         0.095       0.076
##   P-value H_0: RMSEA <= 0.050                    0.004       0.112
##   P-value H_0: RMSEA >= 0.080                    0.435       0.019
##
```

```
##   Robust RMSEA                                                0.079
##   90 Percent confidence interval - lower                      0.056
##   90 Percent confidence interval - upper                      0.102
##   P-value H_0: Robust RMSEA <= 0.050                          0.020
##   P-value H_0: Robust RMSEA >= 0.080                          0.484
##
## Standardized Root Mean Square Residual:
##
##   SRMR                                            0.036       0.036
##
## Parameter Estimates:
##
##   Standard errors                             Sandwich
##   Information bread                           Observed
##   Observed information based on                Hessian
##
## Latent Variables:
##                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##   f1 =~
##     newhabit01        0.899    0.059   15.246    0.000    0.899    0.731
##     newhabit02        1.059    0.043   24.833    0.000    1.059    0.822
##     newhabit03        0.765    0.059   12.889    0.000    0.765    0.629
##     newhabit04        0.889    0.055   16.220    0.000    0.889    0.689
##     newhabit05        0.826    0.053   15.473    0.000    0.826    0.699
##     newhabit06        0.726    0.058   12.445    0.000    0.726    0.652
##   f2 =~
##     newhabit07        0.926    0.062   15.013    0.000    0.926    0.726
##     newhabit08        1.005    0.063   15.936    0.000    1.005    0.721
##     newhabit09        1.015    0.056   18.172    0.000    1.015    0.788
##
## Covariances:
##                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##   f1 ~~
##     f2                0.781    0.041   19.242    0.000    0.781    0.781
##
## Intercepts:
##                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##    .newhabit01        3.404    0.062   55.347    0.000    3.404    2.768
##    .newhabit02        3.228    0.063   51.322    0.000    3.228    2.505
##    .newhabit03        2.908    0.059   49.068    0.000    2.908    2.393
##    .newhabit04        3.083    0.063   49.075    0.000    3.083    2.387
##    .newhabit05        3.264    0.057   56.860    0.000    3.264    2.764
##    .newhabit06        3.531    0.055   64.504    0.000    3.531    3.173
##    .newhabit07        3.516    0.065   54.435    0.000    3.516    2.757
##    .newhabit08        3.276    0.070   46.772    0.000    3.276    2.348
##    .newhabit09        3.214    0.064   50.514    0.000    3.214    2.495
##
## Variances:
##                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##     f1                1.000                               1.000    1.000
##     f2                1.000                               1.000    1.000
##    .newhabit01        0.705    0.086    8.239    0.000    0.705    0.466
##    .newhabit02        0.539    0.062    8.741    0.000    0.539    0.325
##    .newhabit03        0.892    0.091    9.845    0.000    0.892    0.604
```

```
##    .newhabit04        0.877    0.090    9.758    0.000    0.877    0.526
##    .newhabit05        0.713    0.068   10.544    0.000    0.713    0.511
##    .newhabit06        0.712    0.073    9.780    0.000    0.712    0.575
##    .newhabit07        0.769    0.090    8.544    0.000    0.769    0.473
##    .newhabit08        0.936    0.109    8.609    0.000    0.936    0.481
##    .newhabit09        0.629    0.095    6.637    0.000    0.629    0.379
##
## R-Square:
##                    Estimate
##     newhabit01        0.534
##     newhabit02        0.675
##     newhabit03        0.396
##     newhabit04        0.474
##     newhabit05        0.489
##     newhabit06        0.425
##     newhabit07        0.527
##     newhabit08        0.519
##     newhabit09        0.621
```

Scale unidimensionality using omega

```
reliability(twofactorfit)
```

```
##               f1        f2
## alpha  0.8531724 0.7874316
## omega  0.8572908 0.7882256
## omega2 0.8572908 0.7882256
## omega3 0.8609405 0.7889081
## avevar 0.5042199 0.5541094
```

When we run a CFA with these two factors, you'll notice that fit improves compared to the one-factor model, as we accounted for the similarities between items. But you'll also notice that the correlation between the two factors is very high ($r = .78$), which suggests that they have a lot of overlap.

Omega has decreased slightly, but remains high at .86. Also notice that we have two sets of omegas: one for f1 and one for f2.

Given the improvement in fit when we split the measure into two factors, but the extensive overlap between factors, we estimated a bifactor model.

Bifactor CFA Model

The goal here is to examine whether a two-factor structure explains any variability in scores that hasn't been accounted for by the general (habit development) factor.

We have to specify three factors: A general factor (g) and and our two specific factors (f1 and f2)

```
bifactormodel <- 'g =~ newhabit01 + newhabit02 + newhabit03 + newhabit04 +
                      newhabit05 + newhabit06 + newhabit07 + newhabit08 +
                      newhabit09
              f1 =~ newhabit01 + newhabit02 + newhabit03 + newhabit04 +
                    newhabit05 + newhabit06
              f2 =~ newhabit07 + newhabit08 + newhabit09'
```

When fitting the model, all latent variables should be uncorrelated (i.e., orthogonal). To get unstandardized loadings for all items, we can tell lavaan to standardize all factor variances with the code in the cfa fitting function `cfa(..., std.lv = T)` This will automatically free the first indicators of all latent variables.

```
bifactorfit <- cfa(bifactormodel,
                data = habit,
                missing = "fiml",
                estimator = "mlr",
                orthogonal=T, # Making all latent variables uncorrelated
                std.lv=T # Standardized factor variances, freely-estimated loadings
                )
```

```
summary(bifactorfit,
        standardized = TRUE,
        fit.measures = TRUE,
        rsquare = TRUE)
```

```
## lavaan 0.6-20.2265 ended normally after 61 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                        36
##
##                                                   Used       Total
##   Number of observations                           430         432
##   Number of missing patterns                        40
##
## Model Test User Model:
##                                                Standard      Scaled
##   Test Statistic                                 42.020      33.984
##   Degrees of freedom                                 18          18
##   P-value (Chi-square)                            0.001       0.013
##   Scaling correction factor                                   1.236
##     Yuan-Bentler correction (Mplus variant)
##
## Model Test Baseline Model:
##
##   Test statistic                               1554.087    1043.454
##   Degrees of freedom                                 36          36
##   P-value                                         0.000       0.000
##   Scaling correction factor                                   1.489
##
## User Model versus Baseline Model:
##
##   Comparative Fit Index (CFI)                     0.984       0.984
##   Tucker-Lewis Index (TLI)                        0.968       0.968
##
##   Robust Comparative Fit Index (CFI)                          0.987
##   Robust Tucker-Lewis Index (TLI)                             0.974
##
## Loglikelihood and Information Criteria:
##
##   Loglikelihood user model (H0)              -5206.468   -5206.468
```

```
##   Scaling correction factor                                  1.263
##       for the MLR correction
##   Loglikelihood unrestricted model (H1)      -5185.458   -5185.458
##   Scaling correction factor                                  1.254
##       for the MLR correction
##
##   Akaike (AIC)                               10484.936   10484.936
##   Bayesian (BIC)                             10631.233   10631.233
##   Sample-size adjusted Bayesian (SABIC)      10516.990   10516.990
##
## Root Mean Square Error of Approximation:
##
##   RMSEA                                          0.056       0.045
##   90 Percent confidence interval - lower         0.034       0.023
##   90 Percent confidence interval - upper         0.078       0.066
##   P-value H_0: RMSEA <= 0.050                    0.307       0.612
##   P-value H_0: RMSEA >= 0.080                    0.035       0.002
##
##   Robust RMSEA                                               0.053
##   90 Percent confidence interval - lower                     0.018
##   90 Percent confidence interval - upper                     0.083
##   P-value H_0: Robust RMSEA <= 0.050                         0.406
##   P-value H_0: Robust RMSEA >= 0.080                         0.072
##
## Standardized Root Mean Square Residual:
##
##   SRMR                                           0.022       0.022
##
## Parameter Estimates:
##
##   Standard errors                             Sandwich
##   Information bread                           Observed
##   Observed information based on                Hessian
##
## Latent Variables:
##                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##   g =~
##     newhabit01        0.882    0.082   10.725    0.000    0.882    0.717
##     newhabit02        1.050    0.063   16.587    0.000    1.050    0.816
##     newhabit03        0.743    0.070   10.612    0.000    0.743    0.612
##     newhabit04        0.878    0.055   16.067    0.000    0.878    0.680
##     newhabit05        0.833    0.071   11.784    0.000    0.833    0.705
##     newhabit06        0.835    0.072   11.640    0.000    0.835    0.749
##     newhabit07        0.754    0.068   11.119    0.000    0.754    0.591
##     newhabit08        0.737    0.069   10.628    0.000    0.737    0.528
##     newhabit09        0.780    0.063   12.393    0.000    0.780    0.605
##   f1 =~
##     newhabit01        0.304    0.167    1.817    0.069    0.304    0.247
##     newhabit02        0.320    0.271    1.179    0.238    0.320    0.248
##     newhabit03        0.209    0.110    1.901    0.057    0.209    0.172
##     newhabit04        0.103    0.158    0.652    0.514    0.103    0.080
##     newhabit05       -0.114    0.095   -1.195    0.232   -0.114   -0.096
##     newhabit06       -0.611    0.499   -1.224    0.221   -0.611   -0.549
##   f2 =~
```

```
##     newhabit07        0.502    0.099    5.078    0.000    0.502    0.394
##     newhabit08        0.749    0.101    7.417    0.000    0.749    0.537
##     newhabit09        0.640    0.105    6.066    0.000    0.640    0.496
##
## Covariances:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##   g ~~
##     f1                0.000                              0.000    0.000
##     f2                0.000                              0.000    0.000
##   f1 ~~
##     f2                0.000                              0.000    0.000
##
## Intercepts:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##    .newhabit01        3.403    0.061   55.398    0.000    3.403    2.767
##    .newhabit02        3.231    0.063   51.311    0.000    3.231    2.510
##    .newhabit03        2.908    0.059   48.998    0.000    2.908    2.393
##    .newhabit04        3.083    0.063   49.095    0.000    3.083    2.387
##    .newhabit05        3.265    0.057   56.858    0.000    3.265    2.765
##    .newhabit06        3.532    0.055   64.735    0.000    3.532    3.171
##    .newhabit07        3.519    0.064   54.609    0.000    3.519    2.760
##    .newhabit08        3.275    0.070   46.746    0.000    3.275    2.349
##    .newhabit09        3.215    0.064   50.569    0.000    3.215    2.494
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##    .newhabit01        0.643    0.093    6.901    0.000    0.643    0.425
##    .newhabit02        0.452    0.103    4.392    0.000    0.452    0.273
##    .newhabit03        0.880    0.098    8.950    0.000    0.880    0.596
##    .newhabit04        0.887    0.093    9.529    0.000    0.887    0.532
##    .newhabit05        0.688    0.102    6.726    0.000    0.688    0.493
##    .newhabit06        0.170    0.613    0.278    0.781    0.170    0.137
##    .newhabit07        0.805    0.092    8.735    0.000    0.805    0.495
##    .newhabit08        0.840    0.141    5.944    0.000    0.840    0.432
##    .newhabit09        0.643    0.129    4.995    0.000    0.643    0.387
##     g                 1.000                              1.000    1.000
##     f1                1.000                              1.000    1.000
##     f2                1.000                              1.000    1.000
##
## R-Square:
##                    Estimate
##     newhabit01        0.575
##     newhabit02        0.727
##     newhabit03        0.404
##     newhabit04        0.468
##     newhabit05        0.507
##     newhabit06        0.863
##     newhabit07        0.505
##     newhabit08        0.568
##     newhabit09        0.613
```

```
standardizedSolution(bifactorfit, ci = TRUE) # Get CI's if needed
```

```
##           lhs op        rhs est.std    se      z pvalue ci.lower ci.upper
```

```
##  1           g =~ newhabit01   0.717 0.061 11.656  0.000    0.596    0.837
##  2           g =~ newhabit02   0.816 0.043 18.966  0.000    0.732    0.900
##  3           g =~ newhabit03   0.612 0.054 11.277  0.000    0.505    0.718
##  4           g =~ newhabit04   0.680 0.037 18.408  0.000    0.607    0.752
##  5           g =~ newhabit05   0.705 0.053 13.390  0.000    0.602    0.808
##  6           g =~ newhabit06   0.749 0.056 13.330  0.000    0.639    0.860
##  7           g =~ newhabit07   0.591 0.047 12.469  0.000    0.498    0.684
##  8           g =~ newhabit08   0.528 0.046 11.376  0.000    0.437    0.619
##  9           g =~ newhabit09   0.605 0.044 13.747  0.000    0.519    0.692
## 10          f1 =~ newhabit01   0.247 0.136  1.820  0.069   -0.019    0.513
## 11          f1 =~ newhabit02   0.248 0.210  1.180  0.238   -0.164    0.661
## 12          f1 =~ newhabit03   0.172 0.091  1.903  0.057   -0.005    0.350
## 13          f1 =~ newhabit04   0.080 0.122  0.652  0.514   -0.160    0.320
## 14          f1 =~ newhabit05  -0.096 0.080 -1.199  0.231   -0.254    0.061
## 15          f1 =~ newhabit06  -0.549 0.448 -1.223  0.221   -1.427    0.330
## 16          f2 =~ newhabit07   0.394 0.076  5.196  0.000    0.245    0.542
## 17          f2 =~ newhabit08   0.537 0.070  7.648  0.000    0.400    0.675
## 18          f2 =~ newhabit09   0.496 0.081  6.093  0.000    0.337    0.656
## 19 newhabit01 ~~ newhabit01   0.425 0.063  6.781  0.000    0.302    0.548
## 20 newhabit02 ~~ newhabit02   0.273 0.063  4.363  0.000    0.150    0.395
## 21 newhabit03 ~~ newhabit03   0.596 0.061  9.712  0.000    0.476    0.717
## 22 newhabit04 ~~ newhabit04   0.532 0.054  9.941  0.000    0.427    0.637
## 23 newhabit05 ~~ newhabit05   0.493 0.075  6.554  0.000    0.346    0.641
## 24 newhabit06 ~~ newhabit06   0.137 0.493  0.278  0.781   -0.830    1.104
## 25 newhabit07 ~~ newhabit07   0.495 0.057  8.646  0.000    0.383    0.608
## 26 newhabit08 ~~ newhabit08   0.432 0.074  5.858  0.000    0.288    0.577
## 27 newhabit09 ~~ newhabit09   0.387 0.076  5.069  0.000    0.237    0.537
## 28           g ~~           g   1.000 0.000     NA     NA    1.000    1.000
## 29          f1 ~~          f1   1.000 0.000     NA     NA    1.000    1.000
## 30          f2 ~~          f2   1.000 0.000     NA     NA    1.000    1.000
## 31           g ~~          f1   0.000 0.000     NA     NA    0.000    0.000
## 32           g ~~          f2   0.000 0.000     NA     NA    0.000    0.000
## 33          f1 ~~          f2   0.000 0.000     NA     NA    0.000    0.000
## 34 newhabit01 ~1               2.767 0.100 27.646  0.000    2.571    2.964
## 35 newhabit02 ~1               2.510 0.086 29.224  0.000    2.342    2.678
## 36 newhabit03 ~1               2.393 0.075 31.863  0.000    2.246    2.540
## 37 newhabit04 ~1               2.387 0.078 30.546  0.000    2.234    2.540
## 38 newhabit05 ~1               2.765 0.098 28.330  0.000    2.573    2.956
## 39 newhabit06 ~1               3.171 0.132 23.998  0.000    2.912    3.430
## 40 newhabit07 ~1               2.760 0.114 24.253  0.000    2.537    2.983
## 41 newhabit08 ~1               2.349 0.085 27.527  0.000    2.182    2.516
## 42 newhabit09 ~1               2.494 0.092 27.260  0.000    2.315    2.674
## 43           g ~1               0.000 0.000     NA     NA    0.000    0.000
## 44          f1 ~1               0.000 0.000     NA     NA    0.000    0.000
## 45          f2 ~1               0.000 0.000     NA     NA    0.000    0.000
```

Notice that when we fit the bifactor model, all loadings on the general factor fall between .53-.82 and are significantly different from zero, which suggests a general common factor. Moving on to the group factors, the first group factor (items 1-6) is not significant when we account for the general factor.

Scale unidimensionality using omega

We can see the *omega hierarchical* using the reliability function of the semTools package. The first omega is the reliability controlling for the other factors. Here, we can see that the omega for the general factor g is

about .90, and omegas for the specific factors are .01 and .61 after controlling for the general factor's omega.

All omegas provide similar information, but they differ in the different methods used to calculate item-total variances.

```
reliability(bifactorfit) # Omega hierarchical is the third omega. The second omega is based on the mode
```

```
##                g          f1        f2
## alpha  0.8828567 0.853172366 0.7874316
## omega  0.9032614 0.011809432 0.6097963
## omega2 0.8535159 0.001433806 0.3245055
## omega3 0.8518051 0.001436161 0.3248798
## avevar        NA          NA        NA
```

You can also get omega hierarchical for each item using this the Omega_H function of the BifactorIndicesCalculator package (corresponds to the omega based on the observed covariance matrix; the third omega):

```
lambda <- inspect(bifactorfit,what="std")$lambda
theta <- inspect(bifactorfit,what="std")$theta
Omega_H(lambda, theta)
```

```
##            g           f1           f2
## 0.8609644218 0.0005048811 0.3220548248
```

When it comes to omega hierarchical, values above .75 for the general factor and below .50 for the specific factors suggest a dominant general factor and little reliable variance attributable to specific factors. That is because omega hierarchical indexes how much variance in the composite of items is attributable to the each factor (Reise, Bonifay, & Haviland, 2013).

Scale unidimensionaliry using explained common variance (ECV)

The same BifactorIndicesCalculator package will give you ECVs. ECV is a dimensionality index. ECV_SG is the proportion of all common variance explained by that factor. For the general factor, this is simply "ECV," or the proportion of common variance explained by the general factor. For specific factors, the ECV_S computes the strength of a specific factor relative to all explained variance of all items, even those not loading on the specific factor of interest.

(from Dueber, D. M. (2017). Bifactor Indices Calculator: A Microsoft Excel-based tool to calculate various indices relevant to bifactor CFA models. https://dx.doi.org/10.13023/edp.tool.01 [Available at http://sites.education.uky.edu/apslab/resources/]).

You can also get ECV's from Dueber's calculator.

```
ECV_SG(lambda)
```

```
##          g         f1         f2
## 0.77831783 0.08966431 0.13201786
```

ECV_SS is the proportion of all common variance explained by that factor. For the general factor, this is simply "ECV." For specific factors, this ECV_S computes the strength of a specific factor relative to all explained variance only of the items loading on that specific factor.

Because the factors in the bifactor model are orthogonal, the ECV values all sum to 1.0. Values of .70 or higher for the general factor suggest a unidimensional structure (Rodriguez et al., 2016).

```
ECV_SS(lambda)
```

```
##         g        f1        f2
## 0.7783178 0.1323167 0.4095474
```

The omegas, ECV and loadings clearly point to a dominant general factor. The fact that the second factor, though small, doesn't go away when the general factor is modeled is a strong argument against using alpha to index internal consistency.

Practice

How would you specify a model with a higher order factor f3 and two lower-order factors f1 and f2? If you're getting error messages (e.g., negative variance, model is not identified), try to troubleshoot the error. Is it coming from the lower part of the model (with factors f1 and f2 and their indicators) or the upper part (with factors f1, f2, and f3)? Based on what you've estimated so far and what you've seen in lecture, why is the issue coming from _____ part of the model? How can you fix this issue?

Try to answer: * What are the loadings of f1 and f2 on f3? * How do you interpret the meaning of R-square for f1 and f2? How is it different from R-square for the observed items? * Is this a well-fitting model? Why are some indices better than others? (going back to the formulas for each fit index may help) * Consider how fit in nested models refers to the changes in parameters between the two models (recall what modification indices mean, for example). How does the fit between this model and the two correlated-factors model differ? What part of the model does the change in fit refer to?

##References Dueber, D. M. (2017). Bifactor Indices Calculator: A Microsoft Excel-based tool to calculate various indices relevant to bifactor CFA models. https://dx.doi.org/10.13023/edp.tool.01 [Available at http://sites.education.uky.edu/apslab/resources/]

Reise, S. P., Mansolf, M., & Haviland, M. G. (2024). Bifactor measurement models. In R. H. Hoyle (Ed.), Handbook of structural equation modeling (2nd ed). New York: Guilford Press.

Rodriguez, A., Reise, S. P., & Haviland, M. G. (2016). Evaluating bifactor models: Calculating and interpreting statistical indices. Psychological Methods, 21, 137-150. https://doi.org/10.1037/met0000045