Item Analytics of the Grit and Conscientiousness

Nursahid Assafaat

2023-09-11

Contents

View the R Project of this report in this link

A Brief about the Background

This document is intended to analyze the items of Grit and Conscientiousness *(Cons)*. Whether the items already good or not. Special for Grit items, it contains of several items that measure 3 domains: consistency *(CO)*, perseverance *(PE)*, and adaptability *(AD)*.

The steps that will be done to do the item analysis:

- General checking and anomaly
- Cleaned data (remove anomaly, NAs, and fix the formatting if any)
- Item analysis (check item statistics)
  - Reliability analysis of those 4 domains (CO, PE, AD, and Cons) separately.
  - Internal consistency (item-total correlation)
  - Inter-item correlation
  - Confirmatory factor analysis
  - Exploratory factor analysis (as a helper)

```r
dat <- readxl::read_excel("resource/Grit - Studi Kasus.xlsx")
dat <- drop_na(dat)
dat <- dat[6:43]
dim(dat)
```

```
## [1] 215  38
```

```r
names(dat)
```

```
##  [1] "CO1"   "CO2"   "CO3"   "CO4"   "CO5"   "CO6"   "CO7"   "CO8"   "PE1"
## [10] "PE2"   "PE3"   "PE4"   "PE5"   "PE6"   "PE7"   "PE8"   "PE9"   "PE10"
## [19] "PE11"  "AD1"   "AD2"   "AD3"   "AD4"   "AD5"   "AD6"   "AD7"   "AD8"
## [28] "AD9"   "AD10"  "AD11"  "AD12"  "AD13"  "Cons1" "Cons2" "Cons3" "Cons4"
## [37] "Cons5" "Cons6"
```

```r
str(dat)
```

```
## tibble [215 x 38] (S3: tbl_df/tbl/data.frame)
##  $ CO1  : num [1:215] 4 5 5 2 5 5 3 3 4 4 ...
##  $ CO2  : num [1:215] 5 4 4 2 2 4 3 3 5 5 ...
##  $ CO3  : num [1:215] 3 5 4 4 5 5 3 5 5 4 ...
##  $ CO4  : num [1:215] 4 4 1 2 5 3 1 2 4 4 ...
##  $ CO5  : num [1:215] 2 5 5 4 2 5 3 4 5 5 ...
##  $ CO6  : num [1:215] 1 5 4 4 4 4 3 3 5 4 ...
##  $ CO7  : num [1:215] 1 3 2 2 1 5 1 4 1 3 ...
##  $ CO8  : num [1:215] 1 5 1 2 4 2 3 3 4 2 ...
##  $ PE1  : num [1:215] 3 5 1 4 3 3 3 4 5 3 ...
##  $ PE2  : num [1:215] 5 3 4 4 2 5 3 3 5 2 ...
##  $ PE3  : num [1:215] 1 5 4 4 3 5 4 5 5 4 ...
##  $ PE4  : num [1:215] 1 5 4 4 3 2 3 4 5 4 ...
##  $ PE5  : num [1:215] 1 5 2 3 5 2 2 2 4 1 ...
##  $ PE6  : num [1:215] 1 4 2 3 5 4 3 3 4 1 ...
##  $ PE7  : num [1:215] 5 3 4 2 2 5 4 3 4 4 ...
##  $ PE8  : num [1:215] 1 4 4 3 5 5 3 3 5 5 ...
##  $ PE9  : num [1:215] 5 4 3 2 3 1 2 3 5 3 ...
##  $ PE10 : num [1:215] 1 2 3 4 4 4 4 3 5 4 ...
##  $ PE11 : num [1:215] 5 4 2 2 5 5 2 4 1 5 ...
##  $ AD1  : num [1:215] 1 3 4 4 3 4 5 2 5 5 ...
##  $ AD2  : num [1:215] 2 4 5 4 3 4 3 3 4 1 ...
##  $ AD3  : num [1:215] 4 5 5 4 4 5 4 4 3 5 ...
##  $ AD4  : num [1:215] 5 2 2 2 4 3 5 5 3 3 ...
##  $ AD5  : num [1:215] 2 4 5 4 2 3 4 3 5 1 ...
##  $ AD6  : num [1:215] 1 3 5 4 2 2 3 3 4 3 ...
##  $ AD7  : num [1:215] 1 3 5 5 5 4 3 4 3 4 ...
##  $ AD8  : num [1:215] 5 5 1 2 4 5 5 5 5 2 ...
##  $ AD9  : num [1:215] 1 5 1 2 4 5 4 3 2 4 ...
##  $ AD10 : num [1:215] 3 4 5 4 4 3 5 4 5 2 ...
##  $ AD11 : num [1:215] 5 4 5 5 5 5 5 3 5 5 ...
##  $ AD12 : num [1:215] 1 2 1 2 2 5 3 4 3 2 ...
##  $ AD13 : num [1:215] 5 4 1 1 1 4 4 3 3 3 ...
##  $ Cons1: num [1:215] 7 7 7 7 5 7 7 4 7 7 ...
##  $ Cons2: num [1:215] 7 6 7 7 4 6 5 3 5 7 ...
```

```
##  $ Cons3: num [1:215] 6 2 1 1 4 6 5 5 5 3 ...
##  $ Cons4: num [1:215] 7 7 7 7 5 7 7 3 7 7 ...
##  $ Cons5: num [1:215] 6 6 7 7 3 7 4 4 4 7 ...
##  $ Cons6: num [1:215] 7 7 7 7 2 7 6 3 6 7 ...
```

Step 1: General checking and anomaly

I'll check the anomaly using histogram. Since this is ordinal data, I wouldn't expect too much from the shape of histogram. Instead, I'll put my focus more on the value itself.

```
# CO
par(mfrow = c(3,3))
for (i in 1:8) {
  hist(unlist(dat[,i]), main = names(dat[,i]), xlab = "Score")
}
```

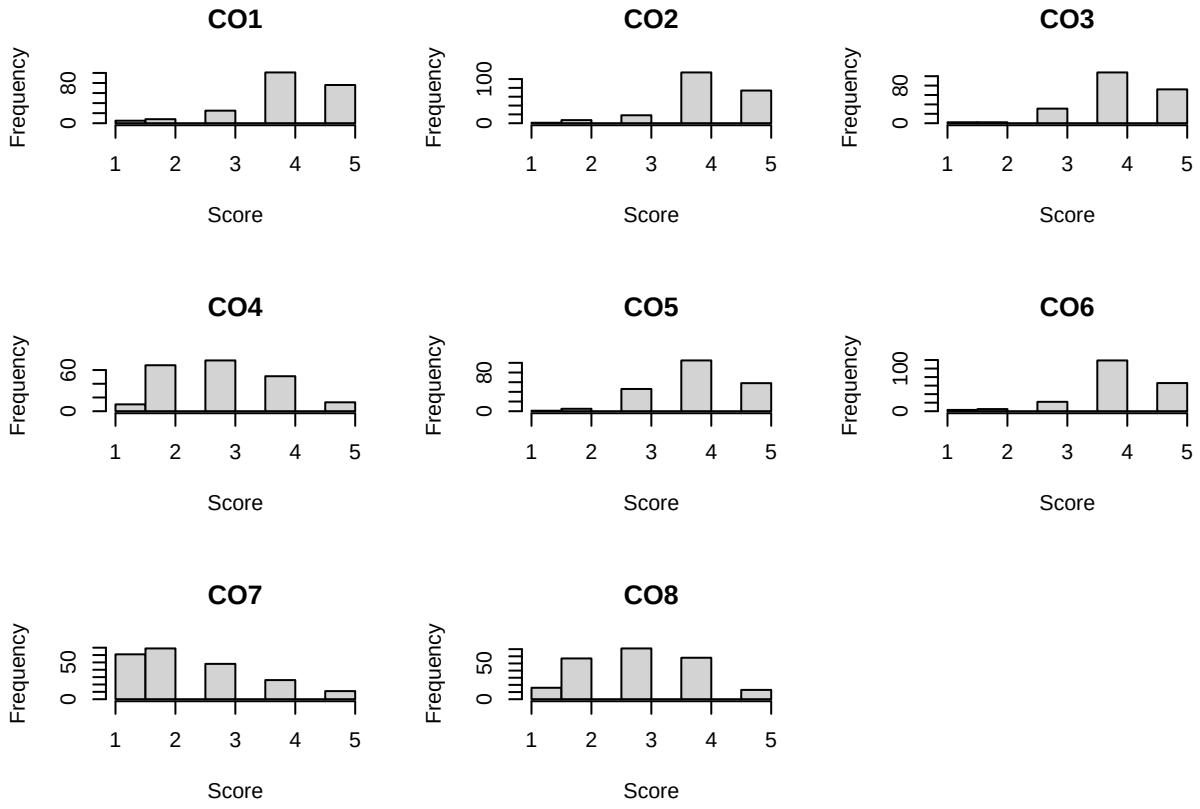


Figure 1: Histograms of Consistency domain

For the consistency domain, everything looks normal. The distribution didn't look very beautiful, but this is due the nature of likert scale.

```
# PE
par(mfrow = c(3,4))
for (i in 9:19) {
  hist(unlist(dat[,i]), main = names(dat[,i]), xlab = "Score")
}
```

Perseverence domain also looks normal. There's no any visible anomaly at this point.

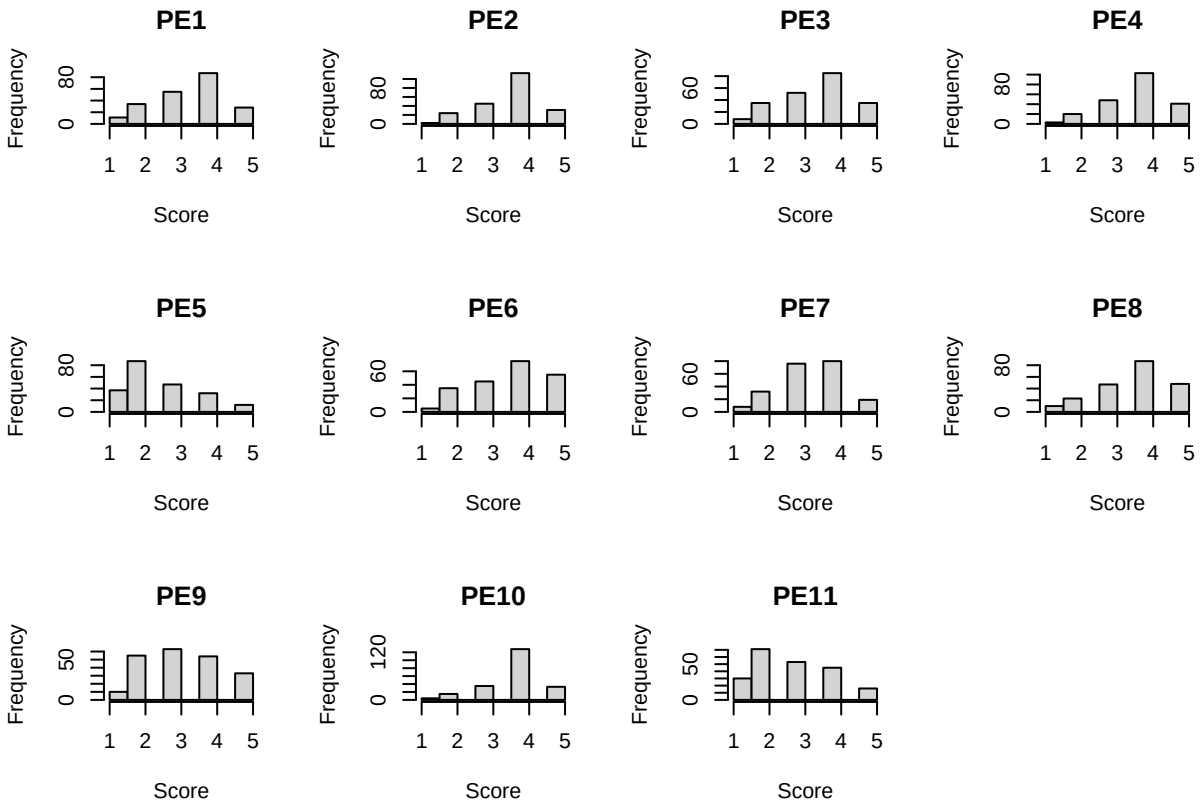


Figure 2: Histograms of Perseverenec domain

```r
# AD
par(mfrow = c(3,5))
for (i in 20:32) {
  hist(unlist(dat[,i]), main = names(dat[,i]), xlab = "Score")
} # AD3 contains 8 value, which not supposed to be since Grit domains contains of likert 1 - 5
```

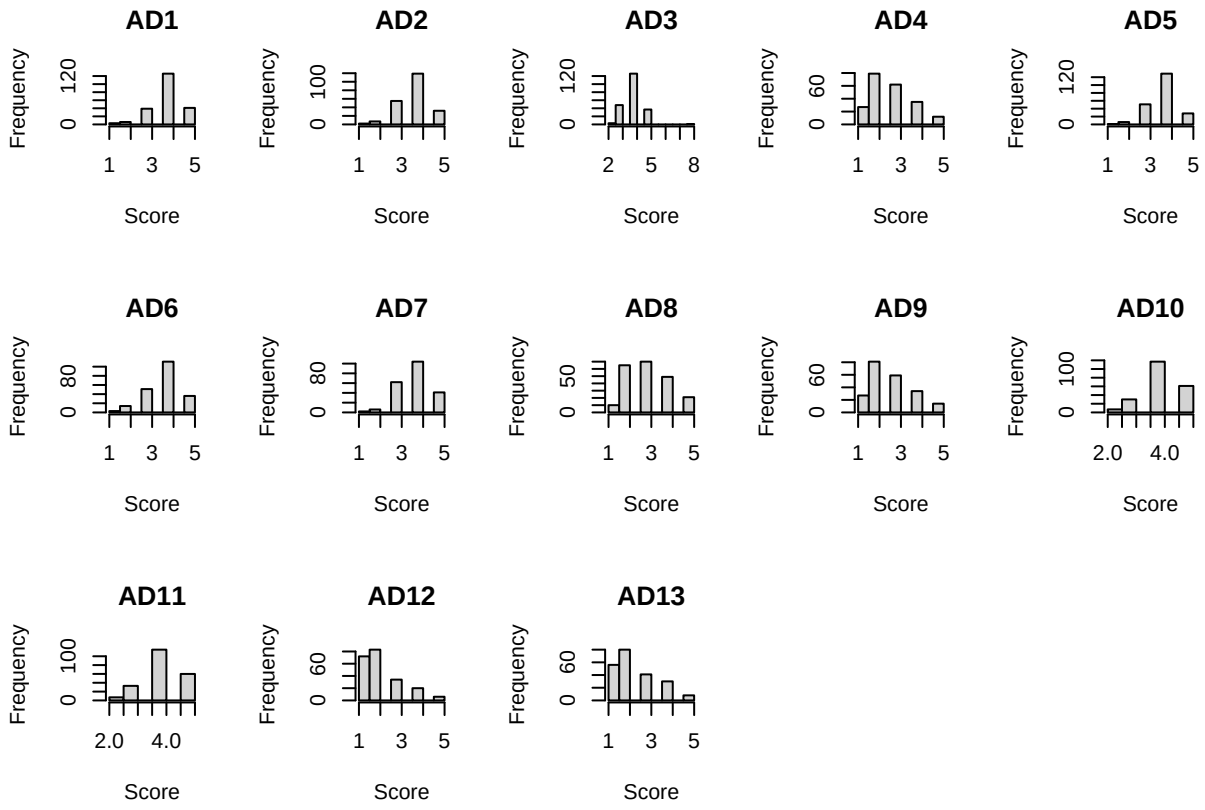


Figure 3: Histograms of Adaptability domain

We can see bit strange thing for this one, the adaptability domain. First, x axis of the histogram of **AD3** have more value, up to 8. Simple checking can shows that indeed there's an 8 value in our data set, which next will be dropped from the data set.

```r
which(dat == 8, arr.ind = TRUE)
```

```
##      row col
## [1,] 106  22
```

Therefore, we can remove row 108 in the cleaning step.

```r
## Cons
par(mfrow = c(2,3))
for (i in 33:38) {
  hist(unlist(dat[,i]), main = names(dat[,i]), xlab = "Score")
}
```

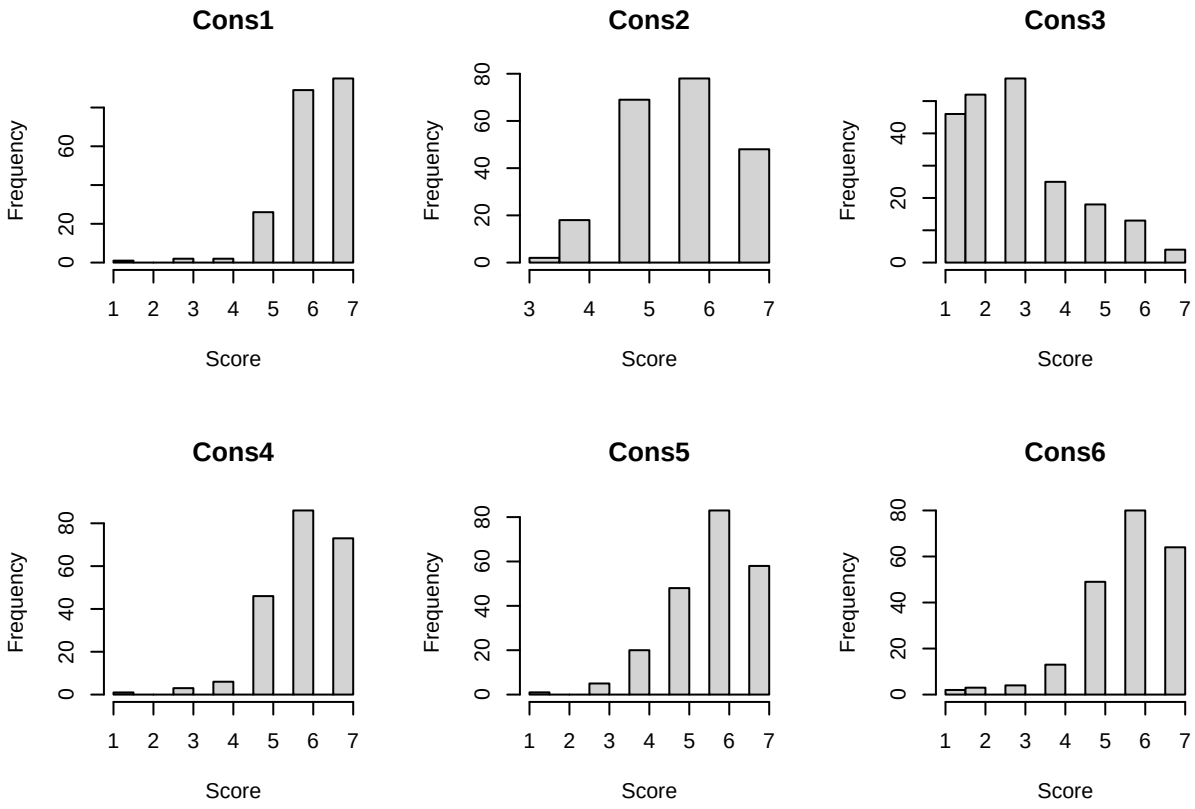


Figure 4: Histograms of Conscientiousness

For the conscientiousness, everything looks normal. The likert scale a bit different, its 1-7 instead of 1-5, but this is no problem et all.

Step 2: Cleaning

In this cleaning process, there are 2 tasks: removing the row that contains an 8 value (108), and removing *NAs*.

```
dat <- dat[-which(dat[,"AD3"] == 8),]
dat <- tidyr::drop_na(dat)
dim(dat)
```

```
## [1] 214  38
```

```
head(dat)
```

```
## # A tibble: 6 x 38
##     CO1   CO2   CO3   CO4   CO5   CO6   CO7   CO8   PE1   PE2   PE3   PE4   PE5
##   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1     4     5     3     4     2     1     1     1     3     5     1     1     1
## 2     5     4     5     4     5     5     3     5     5     3     5     5     5
## 3     5     4     4     1     5     4     2     1     1     4     4     4     2
## 4     2     2     4     2     4     4     2     2     4     4     4     4     3
## 5     5     2     5     5     2     4     1     4     3     2     3     3     5
## 6     5     4     5     3     5     4     5     2     3     5     5     2     2
## # i 25 more variables: PE6 <dbl>, PE7 <dbl>, PE8 <dbl>, PE9 <dbl>, PE10 <dbl>,
## #   PE11 <dbl>, AD1 <dbl>, AD2 <dbl>, AD3 <dbl>, AD4 <dbl>, AD5 <dbl>,
## #   AD6 <dbl>, AD7 <dbl>, AD8 <dbl>, AD9 <dbl>, AD10 <dbl>, AD11 <dbl>,
## #   AD12 <dbl>, AD13 <dbl>, Cons1 <dbl>, Cons2 <dbl>, Cons3 <dbl>, Cons4 <dbl>,
## #   Cons5 <dbl>, Cons6 <dbl>
```

With this task executed, I can proceed to do statistical analysis for the items.

Step 3: Item analysis

Grit Domains

```
cfa <- cfa(
  "CO =~ lambda_1_1*CO1 + lambda_1_2*CO2 + lambda_1_3*CO3 + lambda_1_4*CO4 + lambda_1_5*CO5 + lambda_1_
  PE =~ lambda_2_1*PE1 + lambda_2_2*PE2 + lambda_2_3*PE3 + lambda_2_4*PE4 + lambda_2_5*PE5 + lambda_2_6
  AD =~ lambda_3_1*AD1 + lambda_3_2*AD2 + lambda_3_3*AD3 + lambda_3_4*AD4 + lambda_3_5*AD5 + lambda_3_6
  data = dat,
  estimator = "DWLS"
)

fit_indices <- fitMeasures(cfa)
c(fit_indices["cfi"],
fit_indices["tli"],
```

```
fit_indices["rmsea"],
fit_indices["srmr"]
)
```

```
##        cfi        tli      rmsea       srmr
## 0.73979128 0.72003574 0.08626581 0.11416478
```

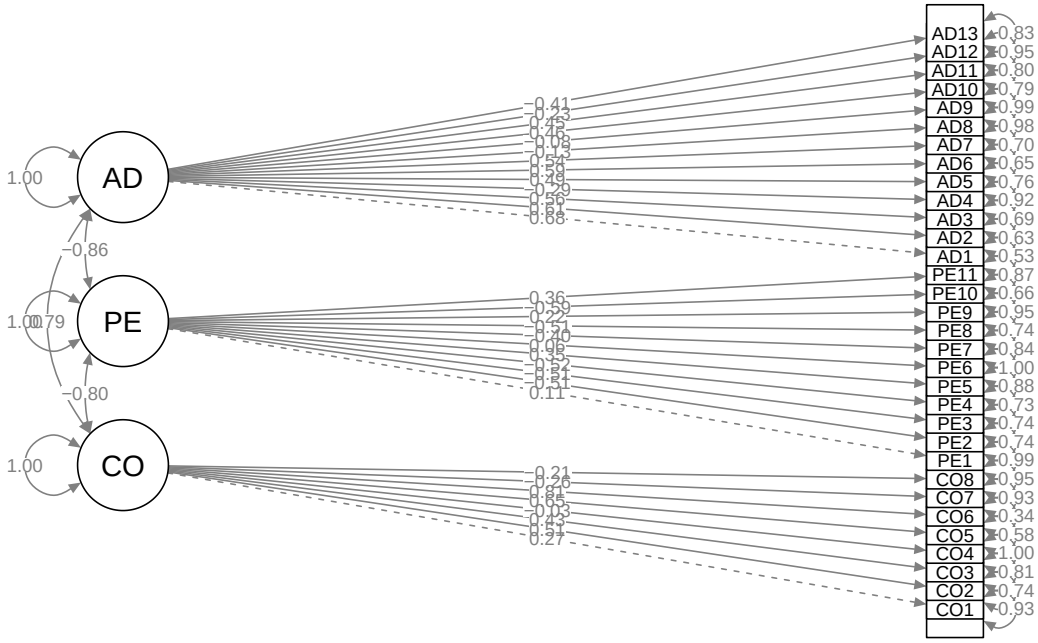
```
summary(cfa, standardized = TRUE)
```

```
## lavaan 0.6.16 ended normally after 103 iterations
##
##   Estimator                                         DWLS
##   Optimization method                           NLMINB
##   Number of model parameters                        67
##
##   Number of observations                           214
##
## Model Test User Model:
##
##   Test statistic                              1191.732
##   Degrees of freedom                               461
##   P-value (Chi-square)                           0.000
##
## Parameter Estimates:
##
##   Standard errors                             Standard
##   Information                                 Expected
##   Information saturated (h1) model        Unstructured
##
## Latent Variables:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##   CO =~
##     CO1    (l_1_1)    1.000                               0.242    0.267
##     CO2    (l_1_2)    1.583    0.243    6.513    0.000    0.383    0.506
##     CO3    (l_1_3)    1.370    0.221    6.205    0.000    0.331    0.433
##     CO4    (l_1_4)   -0.140    0.148   -0.945    0.345   -0.034   -0.034
##     CO5    (l_1_5)    2.103    0.309    6.804    0.000    0.509    0.646
##     CO6    (l_1_6)    2.645    0.382    6.921    0.000    0.640    0.812
##     CO7    (l_1_7)   -1.228    0.228   -5.393    0.000   -0.297   -0.256
##     CO8    (l_1_8)   -0.917    0.189   -4.862    0.000   -0.222   -0.213
##   PE =~
##     PE1   (lm_2_1)    1.000                               0.115    0.108
##     PE2    (l_2_2)   -3.914    1.195   -3.274    0.001   -0.450   -0.506
##     PE3    (l_2_3)   -4.729    1.443   -3.278    0.001   -0.544   -0.513
##     PE4    (l_2_4)   -4.184    1.274   -3.283    0.001   -0.481   -0.522
##     PE5    (l_2_5)    3.342    1.047    3.194    0.001    0.384    0.346
##     PE6    (l_2_6)    0.613    0.376    1.629    0.103    0.071    0.064
##     PE7    (l_2_7)   -3.323    1.032   -3.221    0.001   -0.382   -0.398
##     PE8    (l_2_8)   -4.798    1.468   -3.270    0.001   -0.552   -0.509
##     PE9    (l_2_9)    2.202    0.728    3.024    0.002    0.253    0.224
##     PE10  (l_2_10)   -4.339    1.324   -3.277    0.001   -0.499   -0.586
##     PE11  (l_2_11)    3.584    1.116    3.213    0.001    0.412    0.356
##   AD =~
```

```
##     AD1   (lm_3_1)    1.000                                 0.533    0.684
##     AD2    (l_3_2)    0.871    0.082   10.645    0.000    0.464    0.607
##     AD3    (l_3_3)    0.700    0.065   10.837    0.000    0.373    0.556
##     AD4    (l_3_4)   -0.574    0.079   -7.266    0.000   -0.306   -0.289
##     AD5    (l_3_5)    0.648    0.065    9.926    0.000    0.345    0.492
##     AD6    (l_3_6)    0.944    0.090   10.462    0.000    0.503    0.588
##     AD7    (l_3_7)    0.820    0.081   10.105    0.000    0.437    0.544
##     AD8    (l_3_8)   -0.262    0.070   -3.739    0.000   -0.140   -0.133
##     AD9    (l_3_9)   -0.164    0.069   -2.362    0.018   -0.087   -0.080
##     AD10  (l_3_10)    0.632    0.065    9.766    0.000    0.337    0.462
##     AD11  (l_3_11)    0.628    0.064    9.799    0.000    0.335    0.447
##     AD12  (l_3_12)   -0.453    0.069   -6.518    0.000   -0.241   -0.232
##     AD13  (l_3_13)   -0.866    0.096   -9.012    0.000   -0.461   -0.413
##
## Covariances:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##   CO ~~
##     PE               -0.022    0.007   -3.064    0.002   -0.801   -0.801
##     AD                0.102    0.015    6.708    0.000    0.790    0.790
##   PE ~~
##     AD               -0.053    0.016   -3.293    0.001   -0.860   -0.860
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##    .CO1                0.763    0.112    6.832    0.000    0.763    0.929
##    .CO2                0.426    0.080    5.323    0.000    0.426    0.744
##    .CO3                0.475    0.078    6.062    0.000    0.475    0.812
##    .CO4                0.983    0.079   12.419    0.000    0.983    0.999
##    .CO5                0.361    0.074    4.852    0.000    0.361    0.582
##    .CO6                0.211    0.107    1.973    0.048    0.211    0.340
##    .CO7                1.261    0.115   10.982    0.000    1.261    0.935
##    .CO8                1.030    0.087   11.903    0.000    1.030    0.954
##    .PE1                1.121    0.097   11.615    0.000    1.121    0.988
##    .PE2                0.588    0.083    7.103    0.000    0.588    0.744
##    .PE3                0.829    0.104    7.986    0.000    0.829    0.737
##    .PE4                0.618    0.091    6.817    0.000    0.618    0.728
##    .PE5                1.089    0.108   10.045    0.000    1.089    0.881
##    .PE6                1.210    0.092   13.143    0.000    1.210    0.996
##    .PE7                0.778    0.088    8.833    0.000    0.778    0.842
##    .PE8                0.873    0.119    7.335    0.000    0.873    0.741
##    .PE9                1.211    0.092   13.115    0.000    1.211    0.950
##    .PE10               0.476    0.098    4.859    0.000    0.476    0.656
##    .PE11               1.168    0.104   11.200    0.000    1.168    0.873
##    .AD1                0.323    0.089    3.621    0.000    0.323    0.533
##    .AD2                0.369    0.074    4.954    0.000    0.369    0.631
##    .AD3                0.310    0.046    6.705    0.000    0.310    0.691
##    .AD4                1.029    0.097   10.651    0.000    1.029    0.917
##    .AD5                0.372    0.061    6.070    0.000    0.372    0.758
##    .AD6                0.479    0.087    5.476    0.000    0.479    0.655
##    .AD7                0.454    0.074    6.137    0.000    0.454    0.704
##    .AD8                1.093    0.087   12.547    0.000    1.093    0.982
##    .AD9                1.177    0.100   11.796    0.000    1.177    0.994
##    .AD10               0.419    0.058    7.232    0.000    0.419    0.787
##    .AD11               0.448    0.059    7.563    0.000    0.448    0.800
```

```
##     .AD12              1.020    0.107    9.519    0.000    1.020    0.946
##     .AD13              1.038    0.112    9.305    0.000    1.038    0.830
##      CO                0.059    0.015    3.878    0.000    1.000    1.000
##      PE                0.013    0.008    1.687    0.092    1.000    1.000
##      AD                0.284    0.037    7.722    0.000    1.000    1.000
```

```
semPlot::semPaths(cfa,
                  whatLabels = "std",
                  rotation = 2,
                  sizeMan = 5,
                  node.width = 1,
                  edge.label.cex = .7)
```



From the fit indices and the chi-square, we can see that the model for the Grit hasn't fit yet. Therefore adjustment is required. For easier observation, we can use internal consistency and inter-item correlation

Reliability and Internal Consistency

Consistency domain

```
relCO <- alpha(dat[1:8])
```

```
## Some items ( CO4 CO7 CO8 ) were negatively correlated with the total scale and
```

```
## probably should be reversed.
## To do this, run the function again with the 'check.keys=TRUE' option
```

```
relCO$total$raw_alpha
```

```
## [1] 0.4710277
```

```
relCO$item.stats
```

```
##       n     raw.r     std.r      r.cor     r.drop     mean        sd
## CO1 214 0.5217184 0.5182722 0.37446256 0.28314918 4.098131 0.9062621
## CO2 214 0.4894070 0.5725652 0.51170804 0.28990219 4.182243 0.7564822
## CO3 214 0.6202588 0.6833967 0.66324965 0.44758082 4.144860 0.7645593
## CO4 214 0.5130330 0.4459182 0.28611111 0.24584089 2.953271 0.9918274
## CO5 214 0.4332952 0.5109322 0.45812955 0.21549642 4.000000 0.7872219
## CO6 214 0.4935653 0.5858700 0.57667731 0.28550399 4.116822 0.7874587
## CO7 214 0.3878103 0.2751381 0.05698555 0.04541581 2.331776 1.1616807
## CO8 214 0.3388613 0.2548979 0.04753360 0.03140015 2.971963 1.0387600
```

The internal consistency indicated that CO4, CO7, and CO8 are not eligible for the consistency domain

```
cor(dat[,1:8])
```

```
##            CO1         CO2         CO3         CO4         CO5         CO6
## CO1 1.00000000  0.23401820  0.25041665  0.17748863  0.05264527  0.16148507
## CO2 0.23401820  1.00000000  0.47364704  0.03643257  0.28380955  0.43696618
## CO3 0.25041665  0.47364704  1.00000000  0.08326263  0.35881442  0.43963774
## CO4 0.17748863  0.03643257  0.08326263  1.00000000 -0.03006472 -0.04106673
## CO5 0.05264527  0.28380955  0.35881442 -0.03006472  1.00000000  0.58315788
## CO6 0.16148507  0.43696618  0.43963774 -0.04106673  0.58315788  1.00000000
## CO7 0.08487513 -0.14926215 -0.01736457  0.21725480 -0.03593639 -0.18113882
## CO8 0.03285918 -0.11295840  0.04060669  0.27213608 -0.24687463 -0.14520509
##             CO7         CO8
## CO1  0.08487513  0.03285918
## CO2 -0.14926215 -0.11295840
## CO3 -0.01736457  0.04060669
## CO4  0.21725480  0.27213608
## CO5 -0.03593639 -0.24687463
## CO6 -0.18113882 -0.14520509
## CO7  1.00000000  0.14002582
## CO8  0.14002582  1.00000000
```

```
cor(dat[,1:8]) |> rowSums()
```

```
##       CO1       CO2       CO3       CO4       CO5       CO6       CO7       CO8
## 1.9937881 2.2026530 2.6290206 1.7154433 1.9655514 2.2538362 1.0584538 0.9805897
```

Interitem correlation indicates that CO1 also probably not eligible. To prove that, lets redo the reliability analysis

```
relCO <- alpha(dat[,c(1,2,3,5,6)])
relCO$total$raw_alpha
```

Second reliability analysis after item dropped

```
## [1] 0.6976411
```

```
relCO$item.stats
```

```
##       n     raw.r     std.r     r.cor    r.drop     mean        sd
## CO1 214 0.5348608 0.4998116 0.2765654 0.2292799 4.098131 0.9062621
## CO2 214 0.7034139 0.7145813 0.6089362 0.5114865 4.182243 0.7564822
## CO3 214 0.7331564 0.7422634 0.6497981 0.5516929 4.144860 0.7645593
## CO5 214 0.6607867 0.6704390 0.5763188 0.4412557 4.000000 0.7872219
## CO6 214 0.7632080 0.7713155 0.7269758 0.5893750 4.116822 0.7874587
```

This proves that CO1 also not eligible. Then I will also drop the item.

Perseverence domain

Using the same method, let's see the result

```
relPE <- alpha(dat[9:19])
```

```
## Some items ( PE5 PE9 PE11 ) were negatively correlated with the total scale and
## probably should be reversed.
## To do this, run the function again with the 'check.keys=TRUE' option
```

```
relPE$total$raw_alpha
```

```
## [1] 0.6286576
```

```
relPE$item.stats
```

```
##        n     raw.r     std.r     r.cor    r.drop     mean        sd
## PE1  214 0.4292030 0.4038275 0.2904948 0.2428708 3.406542 1.0650902
## PE2  214 0.3284905 0.3753892 0.2628401 0.1659725 3.682243 0.8891430
## PE3  214 0.5207890 0.5443506 0.4993955 0.3496728 3.490654 1.0603421
## PE4  214 0.5513249 0.5860964 0.5555808 0.4105510 3.742991 0.9214820
## PE5  214 0.5053379 0.4585827 0.4010920 0.3218059 2.514019 1.1121564
## PE6  214 0.5220521 0.4919054 0.4034226 0.3434697 3.649533 1.1020999
## PE7  214 0.5480851 0.5758183 0.5238495 0.3998131 3.322430 0.9611035
## PE8  214 0.5162899 0.5392520 0.4909206 0.3398174 3.649533 1.0849265
## PE9  214 0.3825467 0.3468457 0.2310292 0.1781519 3.210280 1.1290808
## PE10 214 0.3975038 0.4461036 0.3604826 0.2482915 3.799065 0.8512896
## PE11 214 0.3812529 0.3320316 0.2230347 0.1713267 2.742991 1.1564566
```

```r
cor(dat[,9:19])
```

```
##              PE1         PE2         PE3         PE4          PE5         PE6
## PE1   1.00000000 -0.01167560  0.08028597  0.03998929  0.365744255  0.22993776
## PE2  -0.01167560  1.00000000  0.33047493  0.25512318 -0.152148143 -0.06626829
## PE3   0.08028597  0.33047493  1.00000000  0.46120643  0.051866571  0.06347239
## PE4   0.03998929  0.25512318  0.46120643  1.00000000  0.088282086  0.14203474
## PE5   0.36574426 -0.15214814  0.05186657  0.08828209  1.000000000  0.32768813
## PE6   0.22993776 -0.06626829  0.06347239  0.14203474  0.327688134  1.00000000
## PE7   0.05021372  0.24681355  0.25865105  0.45447991  0.037477568  0.13820910
## PE8   0.01418210  0.28309636  0.43585449  0.37908580  0.006036407  0.06170384
## PE9   0.08864253 -0.04536672 -0.08266248 -0.03806028  0.354694608  0.29342172
## PE10 -0.05446511  0.25018953  0.30737661  0.28696712 -0.143295627  0.03968169
## PE11  0.25674709 -0.17567766 -0.13022758 -0.07989766  0.402519218  0.27893650
##             PE7          PE8           PE9          PE10        PE11
## PE1  0.05021372  0.014182099  0.0886425271 -0.0544651140  0.25674709
## PE2  0.24681355  0.283096360 -0.0453667177  0.2501895315 -0.17567766
## PE3  0.25865105  0.435854488 -0.0826624796  0.3073766068 -0.13022758
## PE4  0.45447991  0.379085798 -0.0380602792  0.2869671226 -0.07989766
## PE5  0.03747757  0.006036407  0.3546946083 -0.1432956268  0.40251922
## PE6  0.13820910  0.061703843  0.2934217211  0.0396816920  0.27893650
## PE7  1.00000000  0.343007556  0.0453866166  0.2918687438  0.07068243
## PE8  0.34300756  1.000000000 -0.1081907496  0.4113881695 -0.07586939
## PE9  0.04538662 -0.108190750  1.0000000000  0.0002054219  0.26091302
## PE10 0.29186874  0.411388169  0.0002054219  1.0000000000 -0.11469754
## PE11 0.07068243 -0.075869392  0.2609130177 -0.1146975439  1.00000000
```

```r
cor(dat[,9:19]) |> rowSums()
```

```
##      PE1      PE2      PE3      PE4      PE5      PE6      PE7      PE8
## 2.059602 1.914561 2.776298 2.989211 2.338865 2.508818 2.936790 2.750295
##      PE9     PE10     PE11
## 1.768984 2.275219 1.693428
```

We got item PE5, PE9, and PE11 probably dropped based on the suggestion of reliability analysis. But also, interitem correlation analysis suggest that PE1, and PE6 probably not eligible as well. The result:

```r
relPE <- alpha(dat[c(10,11,12,15,16,18)])
relPE$total$raw_alpha
```

```
## [1] 0.7507333
```

```r
relPE$item.stats
```

```
##        n     raw.r     std.r     r.cor    r.drop     mean        sd
## PE2  214 0.5776520 0.5915877 0.4474363 0.3917286 3.682243 0.8891430
## PE3  214 0.7149008 0.6985837 0.6214655 0.5330077 3.490654 1.0603421
## PE4  214 0.7064721 0.7094115 0.6408248 0.5515627 3.742991 0.9214820
## PE7  214 0.6455176 0.6488843 0.5444130 0.4610147 3.322430 0.9611035
## PE8  214 0.7319017 0.7133050 0.6371198 0.5520224 3.649533 1.0849265
## PE10 214 0.6187914 0.6371234 0.5184875 0.4523782 3.799065 0.8512896
```

```r
cor(dat[,c(10,11,12,15,16,18)])
```

```
##            PE2       PE3       PE4       PE7       PE8      PE10
## PE2  1.0000000 0.3304749 0.2551232 0.2468135 0.2830964 0.2501895
## PE3  0.3304749 1.0000000 0.4612064 0.2586510 0.4358545 0.3073766
## PE4  0.2551232 0.4612064 1.0000000 0.4544799 0.3790858 0.2869671
## PE7  0.2468135 0.2586510 0.4544799 1.0000000 0.3430076 0.2918687
## PE8  0.2830964 0.4358545 0.3790858 0.3430076 1.0000000 0.4113882
## PE10 0.2501895 0.3073766 0.2869671 0.2918687 0.4113882 1.0000000
```

```r
cor(dat[,c(10,11,12,15,16,18)]) |> rowSums()
```

```
##      PE2      PE3      PE4      PE7      PE8     PE10
## 2.365698 2.793563 2.836862 2.594821 2.852432 2.547790
```

Adaptability

```r
relAD <- alpha(dat[20:32])
```

```
## Some items ( AD4 AD8 AD9 AD12 AD13 ) were negatively correlated with the total scale and
## probably should be reversed.
## To do this, run the function again with the 'check.keys=TRUE' option
```

```r
relAD$total$raw_alpha
```

```
## [1] 0.6173245
```

```r
relAD$item.stats
```

```
##        n     raw.r     std.r      r.cor    r.drop     mean        sd
## AD1  214 0.3981007 0.5016623 0.46388040 0.2523521 3.911215 0.7791686
## AD2  214 0.4492126 0.5312448 0.49478435 0.3120248 3.799065 0.7641001
## AD3  214 0.4188964 0.5137264 0.46872607 0.2968424 3.920561 0.6701185
## AD4  214 0.3435550 0.2285341 0.10978242 0.1346069 2.644860 1.0592968
## AD5  214 0.4373721 0.5198417 0.45241509 0.3112108 3.827103 0.7007803
## AD6  214 0.4291840 0.5121221 0.46821700 0.2714717 3.752336 0.8554543
## AD7  214 0.4711362 0.5429494 0.50037163 0.3291543 3.822430 0.8027311
## AD8  214 0.4878982 0.3908606 0.32018426 0.2983740 3.023364 1.0545746
## AD9  214 0.5604788 0.4521010 0.40025622 0.3783739 2.654206 1.0885905
## AD10 214 0.4015333 0.4954283 0.45530179 0.2660996 4.088785 0.7293739
## AD11 214 0.4325674 0.5058607 0.46006310 0.2965465 4.056075 0.7484792
## AD12 214 0.4284409 0.3088434 0.23235528 0.2332459 2.079439 1.0383481
## AD13 214 0.3295306 0.1976621 0.09398586 0.1069027 2.317757 1.1183332
```

With a lot of item and most of the item-total correlation is not vary, we need to rely the analysis to interitem correlation more.

```r
cor(dat[,20:32])
```

```
##               AD1         AD2         AD3         AD4          AD5         AD6
## AD1   1.000000000  0.41937775  0.36407607 -0.21471451  0.324280382  0.45286135
## AD2   0.419377749  1.00000000  0.49130884 -0.13497959  0.311828823  0.40473512
## AD3   0.364076068  0.49130884  1.00000000 -0.11268186  0.340518795  0.26854042
## AD4  -0.214714512 -0.13497959 -0.11268186  1.00000000 -0.076779675 -0.15968820
## AD5   0.324280382  0.31182882  0.34051879 -0.07677967  1.000000000  0.42161702
## AD6   0.452861350  0.40473512  0.26854042 -0.15968820  0.421617020  1.00000000
## AD7   0.485095559  0.27068703  0.18311810 -0.10211621  0.295691248  0.46209169
## AD8  -0.134590554 -0.03493058  0.03585582  0.23020428  0.005491857 -0.07161728
## AD9  -0.008690604  0.04589244  0.01365357  0.32863770 -0.023351580 -0.03189804
## AD10  0.344380459  0.22591361  0.25463468 -0.12306305  0.278174013  0.27618862
## AD11  0.242034068  0.16755573  0.32717254 -0.04582153  0.197585425  0.19043581
## AD12 -0.182737494  0.04979975 -0.14607447  0.38431084 -0.026200016 -0.06231378
## AD13 -0.231477336 -0.18864892 -0.09145234  0.32952727 -0.085323470 -0.23142797
##               AD7          AD8          AD9        AD10        AD11        AD12
## AD1   0.48509556 -0.134590554 -0.008690604  0.34438046  0.24203407 -0.18273749
## AD2   0.27068703 -0.034930576  0.045892437  0.22591361  0.16755573  0.04979975
## AD3   0.18311810  0.035855821  0.013653573  0.25463468  0.32717254 -0.14607447
## AD4  -0.10211621  0.230204279  0.328637696 -0.12306305 -0.04582153  0.38431084
## AD5   0.29569125  0.005491857 -0.023351580  0.27817401  0.19758542 -0.02620002
## AD6   0.46209169 -0.071617278 -0.031898037  0.27618862  0.19043581 -0.06231378
## AD7   1.00000000 -0.044989272  0.047600374  0.36383590  0.31358035 -0.02805762
## AD8  -0.04498927  1.000000000  0.534626414  0.05832737  0.09349870  0.26411957
## AD9   0.04760037  0.534626414  1.000000000 -0.05575875  0.07576811  0.34423482
## AD10  0.36383590  0.058327371 -0.055758751  1.00000000  0.55842706 -0.19532893
## AD11  0.31358035  0.093498696  0.075768109  0.55842706  1.00000000 -0.10845292
## AD12 -0.02805762  0.264119573  0.344234822 -0.19532893 -0.10845292  1.00000000
## AD13 -0.15127111  0.292236384  0.306639688 -0.16137507 -0.12795387  0.46736589
##              AD13
## AD1  -0.23147734
## AD2  -0.18864892
## AD3  -0.09145234
## AD4   0.32952727
## AD5  -0.08532347
## AD6  -0.23142797
## AD7  -0.15127111
## AD8   0.29223638
## AD9   0.30663969
## AD10 -0.16137507
## AD11 -0.12795387
## AD12  0.46736589
## AD13  1.00000000
```

```r
cor(dat[,20:32]) |> rowSums()
```

```
##       AD1       AD2       AD3       AD4       AD5       AD6       AD7       AD8
## 2.859895 3.028540 2.928670 1.302835 2.963533 2.919525 3.095266 2.228233
##       AD9      AD10      AD11      AD12      AD13
## 2.577354 2.824356 2.883829 1.760666 1.126839
```

We have these item possibly dropped: AD4, AD8, AD9, AD12, AD13. And the result:

```
relAD <- alpha(dat[,c(20,21,22,24,25,26,29,30)])
relAD$total$raw_alpha
```

```
## [1] 0.7976335
```

```
relAD$item.stats
```

```
##        n     raw.r     std.r     r.cor    r.drop     mean        sd
## AD1  214 0.7124286 0.7059424 0.6543858 0.5899693 3.911215 0.7791686
## AD2  214 0.6387127 0.6397236 0.5727172 0.4988647 3.799065 0.7641001
## AD3  214 0.6086839 0.6276659 0.5585427 0.4824060 3.920561 0.6701185
## AD5  214 0.6070347 0.6160676 0.5258992 0.4737071 3.827103 0.7007803
## AD6  214 0.6939562 0.6756928 0.6174340 0.5504518 3.752336 0.8554543
## AD7  214 0.6695948 0.6557959 0.5920469 0.5297312 3.822430 0.8027311
## AD10 214 0.6361157 0.6416959 0.5817262 0.5031204 4.088785 0.7293739
## AD11 214 0.5752602 0.5824615 0.5118851 0.4244548 4.056075 0.7484792
```

Final Grit model

After some of the items are dropped, lets check the CFA once again with the new model

```
cfa <- cfa(
  "Factor1 =~ lambda_1_1*CO2 + lambda_1_2*CO3 + lambda_1_3*CO5 + lambda_1_4*CO6
Factor2 =~ lambda_2_1*PE2 + lambda_2_2*PE3 + lambda_2_3*PE4 + lambda_2_4*PE7 + lambda_2_5*PE8 + lambda_
Factor3 =~ lambda_3_1*AD1 + lambda_3_2*AD2 + lambda_3_3*AD3 + lambda_3_4*AD5 + lambda_3_5*AD6 + lambda_
  data = dat,
  estimator = "DWLS"
)

fit_indices <- fitMeasures(cfa)
c(fit_indices["cfi"],
fit_indices["tli"],
fit_indices["rmsea"],
fit_indices["srmr"]
)
```

```
##        cfi        tli      rmsea       srmr
## 1.00000000 1.00240447 0.00000000 0.07300567
```

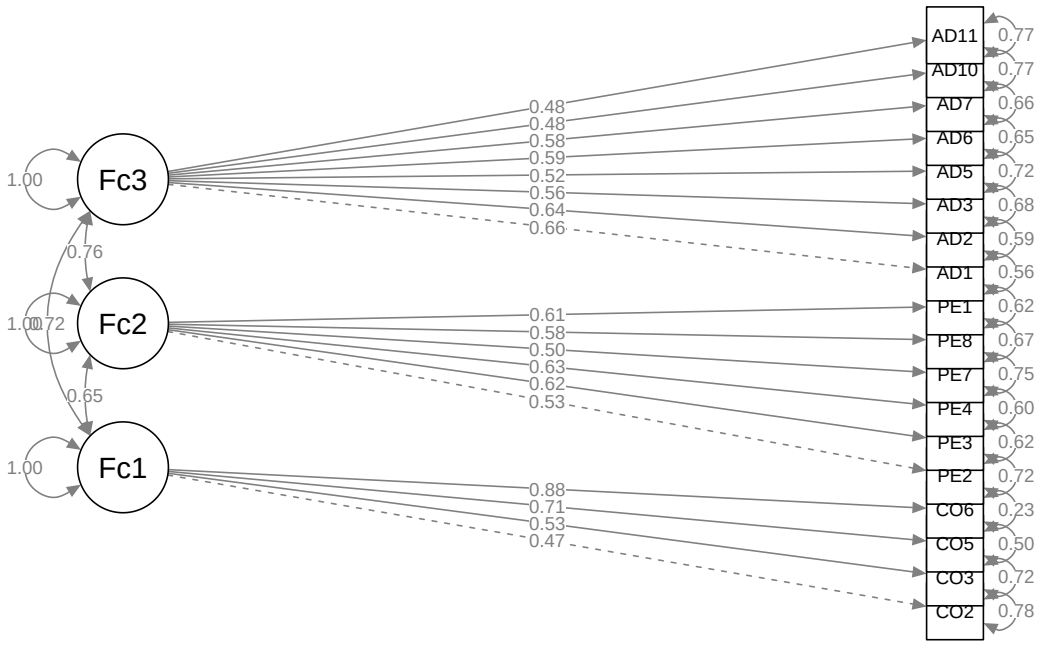
```
summary(cfa, standardized = TRUE)
```

```
## lavaan 0.6.16 ended normally after 50 iterations
##
##   Estimator                                         DWLS
##   Optimization method                           NLMINB
##   Number of model parameters                        39
##
##   Number of observations                           214
##
## Model Test User Model:
```

```
##
##   Test statistic                                128.566
##   Degrees of freedom                                132
##   P-value (Chi-square)                            0.568
##
## Parameter Estimates:
##
##   Standard errors                             Standard
##   Information                                 Expected
##   Information saturated (h1) model        Unstructured
##
## Latent Variables:
##                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##   Factor1 =~
##     CO2    (l_1_1)    1.000                               0.354    0.468
##     CO3    (l_1_2)    1.143    0.156    7.316    0.000    0.404    0.529
##     CO5    (l_1_3)    1.581    0.199    7.923    0.000    0.559    0.710
##     CO6    (l_1_4)    1.948    0.242    8.053    0.000    0.689    0.875
##   Factor2 =~
##     PE2    (l_2_1)    1.000                               0.473    0.532
##     PE3    (l_2_2)    1.385    0.148    9.331    0.000    0.655    0.618
##     PE4    (l_2_3)    1.227    0.130    9.425    0.000    0.581    0.630
##     PE7    (l_2_4)    1.025    0.119    8.626    0.000    0.485    0.504
##     PE8    (l_2_5)    1.327    0.147    9.014    0.000    0.628    0.579
##     PE10   (l_2_6)    1.104    0.125    8.856    0.000    0.522    0.613
##   Factor3 =~
##     AD1    (l_3_1)    1.000                               0.518    0.665
##     AD2    (l_3_2)    0.945    0.096    9.820    0.000    0.489    0.640
##     AD3    (l_3_3)    0.730    0.074    9.919    0.000    0.378    0.564
##     AD5    (l_3_4)    0.710    0.076    9.356    0.000    0.368    0.525
##     AD6    (l_3_5)    0.977    0.102    9.570    0.000    0.506    0.592
##     AD7    (l_3_6)    0.901    0.095    9.463    0.000    0.467    0.581
##     AD10   (l_3_7)    0.679    0.075    9.044    0.000    0.352    0.482
##     AD11   (l_3_8)    0.693    0.075    9.229    0.000    0.359    0.479
##
## Covariances:
##                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##   Factor1 ~~
##     Factor2           0.108    0.015    7.131    0.000    0.647    0.647
##     Factor3           0.132    0.017    7.544    0.000    0.719    0.719
##   Factor2 ~~
##     Factor3           0.186    0.021    8.942    0.000    0.759    0.759
##
## Variances:
##                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##    .CO2               0.447    0.080    5.578    0.000    0.447    0.781
##    .CO3               0.421    0.081    5.168    0.000    0.421    0.720
##    .CO5               0.307    0.080    3.817    0.000    0.307    0.496
##    .CO6               0.145    0.115    1.262    0.207    0.145    0.234
##    .PE2               0.567    0.085    6.700    0.000    0.567    0.717
##    .PE3               0.695    0.112    6.216    0.000    0.695    0.618
##    .PE4               0.512    0.096    5.321    0.000    0.512    0.603
##    .PE7               0.689    0.092    7.459    0.000    0.689    0.746
##    .PE8               0.783    0.125    6.275    0.000    0.783    0.665
```

```
##     .PE10             0.452    0.100    4.513    0.000    0.452    0.624
##     .AD1              0.339    0.090    3.760    0.000    0.339    0.558
##     .AD2              0.344    0.076    4.507    0.000    0.344    0.590
##     .AD3              0.306    0.047    6.530    0.000    0.306    0.682
##     .AD5              0.356    0.062    5.724    0.000    0.356    0.725
##     .AD6              0.476    0.089    5.365    0.000    0.476    0.650
##     .AD7              0.426    0.076    5.621    0.000    0.426    0.662
##     .AD10             0.408    0.059    6.963    0.000    0.408    0.768
##     .AD11             0.432    0.060    7.179    0.000    0.432    0.770
##      Factor1          0.125    0.027    4.680    0.000    1.000    1.000
##      Factor2          0.224    0.036    6.241    0.000    1.000    1.000
##      Factor3          0.268    0.039    6.929    0.000    1.000    1.000
```

```
semPlot::semPaths(cfa,
                  whatLabels = "std",
                  rotation = 2,
                  sizeMan = 5,
                  node.width = 1,
                  edge.label.cex = .7)
```



Now we have a better fit model compared to previous one. Let's proceed to the next one, conscientiousness.

Conscientiousness

```
relCons <- alpha(dat[33:38])
```

```
## Some items ( Cons3 ) were negatively correlated with the total scale and
## probably should be reversed.
## To do this, run the function again with the 'check.keys=TRUE' option
```

```
relCons$total$raw_alpha
```

```
## [1] 0.4548687
```

```
relCons$item.stats
```

```
##         n       raw.r      std.r      r.cor     r.drop     mean        sd
## Cons1 214  0.68551854  0.7458642  0.7164349  0.5166415 6.261682 0.8484888
## Cons2 214  0.66766546  0.7042985  0.5986110  0.4687273 5.710280 0.9395768
## Cons3 214 -0.05893282 -0.2118704 -0.5049390 -0.4521681 2.859813 1.5409807
## Cons4 214  0.76795914  0.8320312  0.8649651  0.6101195 6.014019 0.9468454
## Cons5 214  0.68526631  0.7105359  0.6456994  0.4576299 5.771028 1.0698172
## Cons6 214  0.74515243  0.7591132  0.7020718  0.5211170 5.794393 1.1647266
```

```
cor(dat[,33:38])
```

```
##            Cons1      Cons2      Cons3      Cons4      Cons5      Cons6
## Cons1  1.0000000  0.3899953 -0.3272891  0.6966681  0.3559547  0.5250101
## Cons2  0.3899953  1.0000000 -0.2584065  0.5428680  0.4101017  0.4086389
## Cons3 -0.3272891 -0.2584065  1.0000000 -0.4459056 -0.3726927 -0.3457214
## Cons4  0.6966681  0.5428680 -0.4459056  1.0000000  0.5871696  0.5645676
## Cons5  0.3559547  0.4101017 -0.3726927  0.5871696  1.0000000  0.5347446
## Cons6  0.5250101  0.4086389 -0.3457214  0.5645676  0.5347446  1.0000000
```

```
cor(dat[,33:38]) |> rowSums()
```

```
##      Cons1      Cons2      Cons3      Cons4      Cons5      Cons6
##  2.6403390  2.4931974 -0.7500153  2.9453676  2.5152778  2.6872398
```

This is shown already that Cons3 is the impostor. Let's drop it.

```
relCons <- alpha(dat[c(33,34,36,37,38)])
relCons$total$raw_alpha
```

```
## [1] 0.8298766
```

```
relCons$item.stats
```

```
##         n     raw.r     std.r     r.cor    r.drop     mean        sd
## Cons1 214 0.7432571 0.7654366 0.7082299 0.6160959 6.261682 0.8484888
## Cons2 214 0.6997861 0.7097177 0.5875026 0.5382225 5.710280 0.9395768
## Cons4 214 0.8643053 0.8747069 0.8658472 0.7760096 6.014019 0.9468454
## Cons5 214 0.7611710 0.7448906 0.6579828 0.5979402 5.771028 1.0698172
## Cons6 214 0.8039030 0.7822879 0.7001892 0.6449524 5.794393 1.1647266
```

Now all the items are good.

Summary

In the end, here's the item last that fit to the model.

- CO: CO2, CO3, CO5, CO6
- PE: PE2, PE3, PE4, PE7, PE8, PE10
- AD: AD1, AD2, AD3, AD5, AD6, AD7, AD10, AD11
- Cons: Cons1, Cons2, Cons4, Cons5, Cons6