The reliability and validity of the compulsory citizenship behaviors scale: Six-Step R-based psychometrics protocol among nurses in Turkey

Bora YILDIZ

2/13/2021

```
###################################
#
# R source code for data analysis in
# "RELIABILITY AND VALIDITY OF THE COMPULSORY CITIZENSHIP BEHAVIORS SCALE:
#  6-STEP R-BASED PSYCHOMETRICS PROTOCOL AMONG NURSES IN TURKEY"
#
###################################

# Packages
for (n in c('readxl','haven','GPArotation', 'corrplot', 'ggpubr', 'SparseM', 'foreign', 'utils', 'relimp
            'ggplot2', 'ggdendro', 'psych', 'Hmisc', 'ltm', 'mirt', 'eRm', 'mokken', 'lavaan','OpenMx',
            'semTools','semPlot', 'qgraph','sem','CTT','pwr','PerformanceAnalytics','RColorBrewer','ggc
            'GGally','corrr','RColorBrewer','MBESS','cluster')) {if(!require(n,character.only=TRUE)){ins
```

```
## Loading required package: readxl
```

```
## Loading required package: haven
```

```
## Loading required package: GPArotation
```

```
## Loading required package: corrplot
```

```
## corrplot 0.84 loaded
```

```
## Loading required package: ggpubr
```

```
## Loading required package: ggplot2
```

```
## Loading required package: SparseM
```

```
##
## Attaching package: 'SparseM'
```

```
## The following object is masked from 'package:base':
##
##     backsolve
```

```
## Loading required package: foreign

## Loading required package: relimp

## Loading required package: ggdendro

## Loading required package: psych

##
## Attaching package: 'psych'

## The following objects are masked from 'package:ggplot2':
##
##     %+%, alpha

## Loading required package: Hmisc

## Loading required package: lattice

## Loading required package: survival

## Loading required package: Formula

##
## Attaching package: 'Hmisc'

## The following object is masked from 'package:psych':
##
##     describe

## The following object is masked from 'package:ggdendro':
##
##     label

## The following objects are masked from 'package:base':
##
##     format.pval, units

## Loading required package: ltm

## Loading required package: MASS

## Loading required package: msm

## Loading required package: polycor

##
## Attaching package: 'polycor'
```

```
## The following object is masked from 'package:psych':
##
##     polyserial

##
## Attaching package: 'ltm'

## The following object is masked from 'package:psych':
##
##     factor.scores

## Loading required package: mirt

## Loading required package: stats4

##
## Attaching package: 'mirt'

## The following object is masked from 'package:ltm':
##
##     Science

## Loading required package: eRm

##
## Attaching package: 'eRm'

## The following objects are masked from 'package:mirt':
##
##     itemfit, personfit

## The following object is masked from 'package:psych':
##
##     sim.rasch

## Loading required package: mokken

## Loading required package: poLCA

## Loading required package: scatterplot3d

##
## Attaching package: 'mokken'

## The following object is masked from 'package:psych':
##
##     ICC

## Loading required package: lavaan
```

```
## This is lavaan 0.6-7

## lavaan is BETA software! Please report any bugs.

##
## Attaching package: 'lavaan'

## The following object is masked from 'package:psych':
##
##     cor2cov

## Loading required package: OpenMx

## To take full advantage of multiple cores, use:
##   mxOption(key='Number of Threads', value=parallel::detectCores()) #now
##   Sys.setenv(OMP_NUM_THREADS=parallel::detectCores()) #before library(OpenMx)

##
## Attaching package: 'OpenMx'

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

## The following object is masked from 'package:psych':
##
##     tr

## Loading required package: writexl

## Loading required package: semTools

##

## ################################################################################

## This is semTools 0.5-4

## All users of R (or SEM) are invited to submit functions or ideas for functions.

## ################################################################################

##
## Attaching package: 'semTools'

## The following object is masked from 'package:psych':
##
##     skew
```

```
## Loading required package: semPlot

## Registered S3 methods overwritten by 'lme4':
##   method                          from
##   cooks.distance.influence.merMod car
##   influence.merMod                car
##   dfbeta.influence.merMod         car
##   dfbetas.influence.merMod        car

## Loading required package: qgraph

## Loading required package: sem

##
## Attaching package: 'sem'

## The following object is masked from 'package:OpenMx':
##
##     Bollen

## The following objects are masked from 'package:lavaan':
##
##     cfa, sem

## The following object is masked from 'package:mirt':
##
##     fscores

## Loading required package: CTT

##
## Attaching package: 'CTT'

## The following object is masked from 'package:semTools':
##
##     reliability

## The following object is masked from 'package:polycor':
##
##     polyserial

## The following object is masked from 'package:psych':
##
##     polyserial

## Loading required package: pwr

## Loading required package: PerformanceAnalytics

## Loading required package: xts
```

```
## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

##
## Attaching package: 'PerformanceAnalytics'

## The following object is masked from 'package:semTools':
##
##     kurtosis

## The following object is masked from 'package:graphics':
##
##     legend

## Loading required package: RColorBrewer

## Loading required package: ggcorrplot

## Loading required package: GGally

## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2

## Loading required package: corrr

## Loading required package: MBESS

##
## Attaching package: 'MBESS'

## The following object is masked from 'package:lavaan':
##
##     cor2cov

## The following object is masked from 'package:psych':
##
##     cor2cov

## Loading required package: cluster
```

```
library(n,character.only=TRUE)

# Reading the data
mydata <- read_sav("data(ccb_530_noout).sav")
names(mydata)
```

```
##  [1] "Age"             "Age_Categorical" "Gender"          "Education"
##  [5] "Sector"          "ccb_1"           "ccb_2"           "ccb_3"
##  [9] "ccb_4"           "ccb_5"
```

```
dim(mydata)
```

```
## [1] 530  10
```

```
#####################  STEP 1 - Descriptive statistics ######################
# Sum of all WEMWBS items
attach(mydata)
CCB_total <- ccb_1 + ccb_2 + ccb_3 + ccb_4 + ccb_5
mydata$CCB_total <- CCB_total
detach(mydata)
summary(mydata$CCB_total)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    5.00   13.00   18.00   16.74   20.00   25.00
```

```
CCB_table <- mydata[,6:10]
lowerCor(CCB_table, method = "spearman")
```

```
##       ccb_1 ccb_2 ccb_3 ccb_4 ccb_5
## ccb_1 1.00
## ccb_2 0.85  1.00
## ccb_3 0.77  0.81  1.00
## ccb_4 0.70  0.74  0.77  1.00
## ccb_5 0.60  0.63  0.65  0.83  1.00
```

```
pairs.panels(CCB_table, scale=TRUE, stars=T, ci=TRUE, alpha=0.05, digits = 3, # Correlation chart (pear
             method = "pearson", hist.col = "green",cex.cor=1)
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : pseudoinverse used at 3

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : neighborhood radius 1
```

```
## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : reciprocal condition
## number 0

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : There are other near
## singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : pseudoinverse used at 3

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : neighborhood radius 1

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : reciprocal condition
## number 0

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : There are other near
## singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : pseudoinverse used at 3
```

```
## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : neighborhood radius 1

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : reciprocal condition
## number 0

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : There are other near
## singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : pseudoinverse used at 3

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : neighborhood radius 1

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : reciprocal condition
## number 0

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : There are other near
## singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : pseudoinverse used at 4

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : neighborhood radius 1

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : reciprocal condition
## number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : pseudoinverse used at 4

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : neighborhood radius 1
```

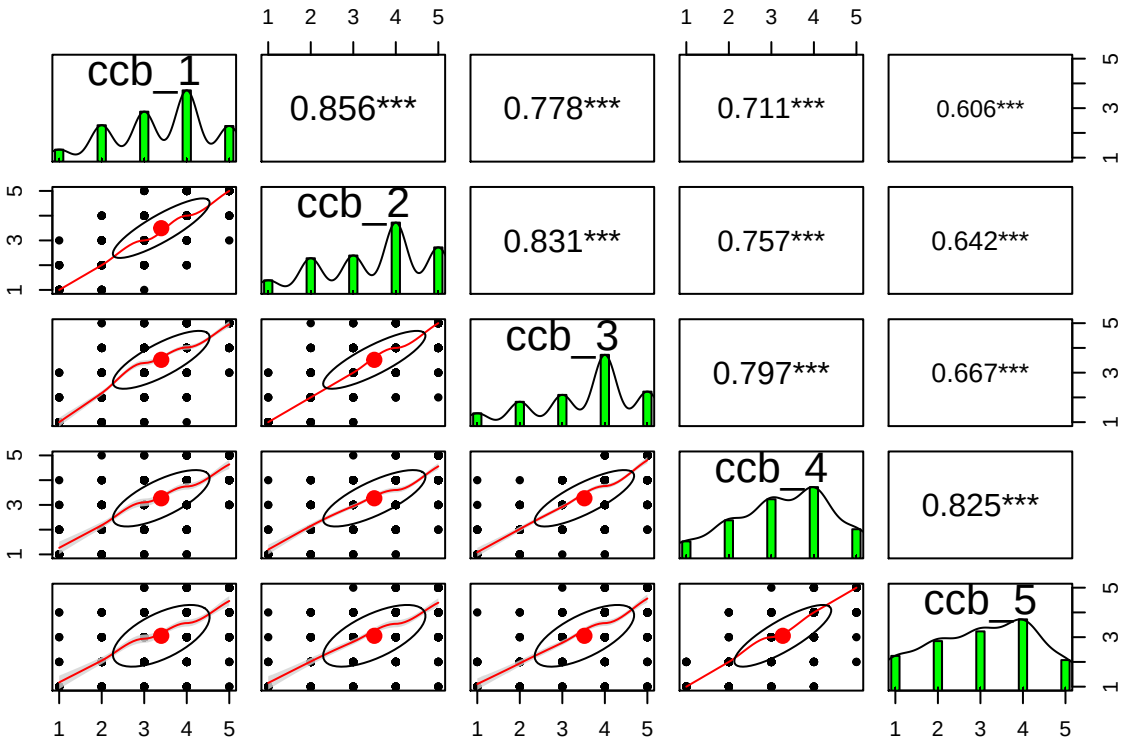
```
## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : reciprocal condition
## number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0
```

```
## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : pseudoinverse used at 4

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : neighborhood radius 1

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : reciprocal condition
## number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : pseudoinverse used at 4

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : neighborhood radius 1

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : reciprocal condition
## number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : pseudoinverse used at 4

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : neighborhood radius 1

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : reciprocal condition
## number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number -0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number -0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number -0
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number -0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number -0

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : pseudoinverse used at 3

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : neighborhood radius 1

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : reciprocal condition
## number -0
```



```r
pairs.panels(CCB_table, scale=TRUE, stars=T, ci=TRUE, alpha=0.05, digits = 3, # Correlation chart (Spea
             method = "spearman",hist.col = "green", cex.cor=1)
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : pseudoinverse used at 3

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : neighborhood radius 1

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : reciprocal condition
## number 0

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : There are other near
## singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : pseudoinverse used at 3

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : neighborhood radius 1

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : reciprocal condition
## number 0
```

```
## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : There are other near
## singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : pseudoinverse used at 3

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : neighborhood radius 1

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : reciprocal condition
## number 0

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : There are other near
## singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : pseudoinverse used at 3

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : neighborhood radius 1
```

```
## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : reciprocal condition
## number 0

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : There are other near
## singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : pseudoinverse used at 4

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : neighborhood radius 1

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : reciprocal condition
## number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0
```

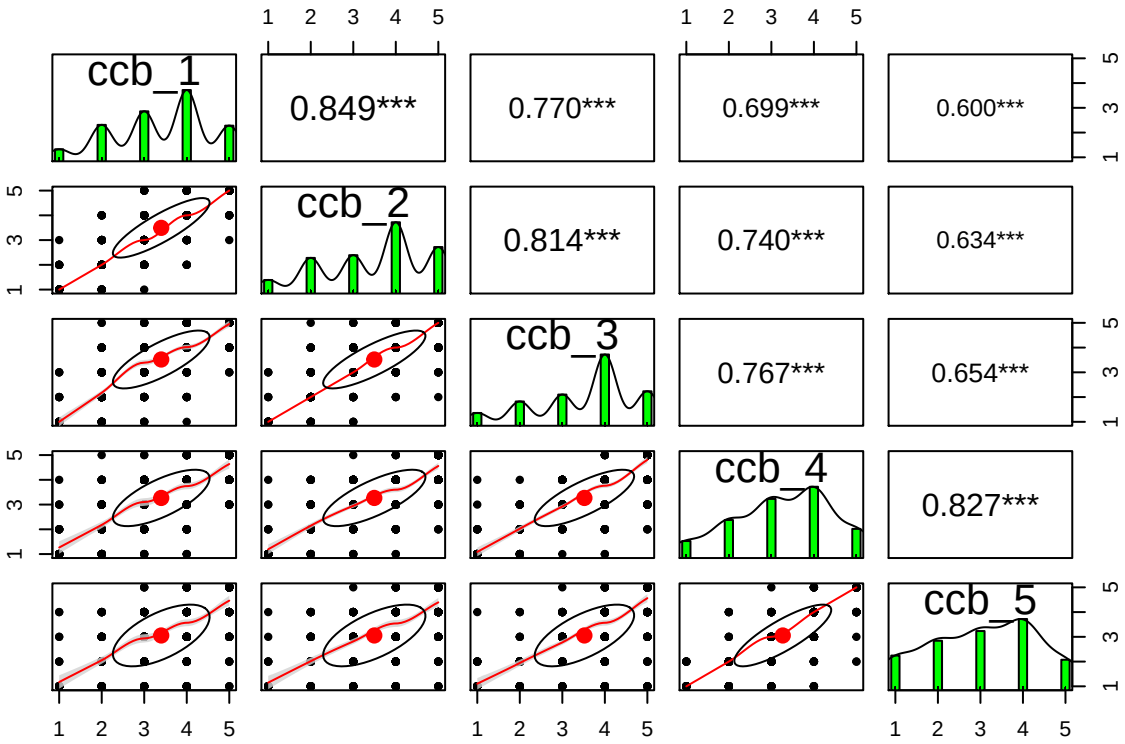
```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : pseudoinverse used at 4

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : neighborhood radius 1

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : reciprocal condition
## number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : pseudoinverse used at 4

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : neighborhood radius 1

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : reciprocal condition
## number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : pseudoinverse used at 4

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : neighborhood radius 1

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : reciprocal condition
## number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : pseudoinverse used at 4

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : neighborhood radius 1

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : reciprocal condition
## number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number -0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number -0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number -0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number -0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number -0

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : pseudoinverse used at 3

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : neighborhood radius 1

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : reciprocal condition
## number -0
```




```
#Power analysis
correlation <- c(.849,.770, .814,.699,.740,.767,.600,.634,.654,.827)
mean(correlation)
```

```
## [1] 0.7354
```

```
pwr.r.test(n =530, r = .735, sig.level = 0.05, power =)
```

```
##
##      approximate correlation power calculation (arctangh transformation)
##
##               n = 530
##               r = 0.735
##       sig.level = 0.05
##           power = 1
##     alternative = two.sided
```

```
###################### STEP 2 - IRT analyses (the mokken, ltm, and mirt packages) ######################
moscales.for.lowerbounds <- function( x, lowerbounds=seq(from=0.05,to=0.60,by=0.05) )
{
  ret.value <- NULL;
  for( lowerbound in lowerbounds )
  {
    tmp <- aisp( x,  lowerbound=lowerbound );
    if( is.null(ret.value) )
    {
      ret.value <- data.frame( "Item"=rownames(tmp), "Scales."=tmp[,1] );
    }
```

```
    else
    {
      ret.value <- cbind( ret.value, "Scales."=tmp[,1] );
    }
    names(ret.value)[ncol(ret.value)] <- paste("c=",sprintf("%.2f",lowerbound),sep="");
  }
  rownames(ret.value) <- NULL;
  ret.value;
}

# Compute scalability coefficients
CCB_table <- as.data.frame(CCB_table[complete.cases(CCB_table),])
coefH(CCB_table)$H
```

```
## $Hij
##       ccb_1   se      ccb_2   se      ccb_3   se      ccb_4   se      ccb_5
## ccb_1                  0.890  (0.015)  0.819  (0.022)  0.744  (0.026)  0.656
## ccb_2  0.890  (0.015)                  0.860  (0.018)  0.808  (0.023)  0.698
## ccb_3  0.819  (0.022)  0.860  (0.018)                  0.857  (0.020)  0.727
## ccb_4  0.744  (0.026)  0.808  (0.023)  0.857  (0.020)                  0.873
## ccb_5  0.656  (0.033)  0.698  (0.032)  0.727  (0.032)  0.873  (0.018)
##       se
## ccb_1 (0.033)
## ccb_2 (0.032)
## ccb_3 (0.032)
## ccb_4 (0.018)
## ccb_5
##
## $Hi
##       Item H  se
## ccb_1   0.777 (0.019)
## ccb_2   0.813 (0.016)
## ccb_3   0.815 (0.017)
## ccb_4   0.821 (0.016)
## ccb_5   0.739 (0.025)
##
## $H
## Scale H      se
##   0.793 (0.017)

## Scale H      se
##   0.793 (0.017)
```

```
# examine aisp for increasing c levels (run the function you defined above and give it a name)
motable.CCB_table <- moscales.for.lowerbounds( CCB_table )

# see the results
motable.CCB_table
```

```
##    Item c=0.05 c=0.10 c=0.15 c=0.20 c=0.25 c=0.30 c=0.35 c=0.40 c=0.45 c=0.50
## 1 ccb_1      1      1      1      1      1      1      1      1      1      1
## 2 ccb_2      1      1      1      1      1      1      1      1      1      1
```

```
## 3 ccb_3      1      1      1      1      1      1      1      1      1      1
## 4 ccb_4      1      1      1      1      1      1      1      1      1      1
## 5 ccb_5      1      1      1      1      1      1      1      1      1      1
##   c=0.55 c=0.60
## 1      1      1
## 2      1      1
## 3      1      1
## 4      1      1
## 5      1      1
```

```
# save it as a data frame
CCB_table2 <- as.data.frame(motable.CCB_table)

###################### STEP 3 - Parametric IRT ######################
# Rating Scale model (equivalent of Rasch for ordinal items)

fit1.CCB_table2 <- PCM(CCB_table) #, constrained = FALSE, Hessian=TRUE
```

```
## Warning:
## The following items have no 0-responses:
## ccb_1 ccb_2 ccb_3 ccb_4 ccb_5
## Responses are shifted such that lowest category is 0.
```

```
# separation reliability (proportion of item variance not due to error - similar to C-alpha)
ppr1 <- person.parameter(fit1.CCB_table2)

# item fit (between 0.6 and 1.4 acc to Wright BD, Linacre JM. Reasonable mean-square fit values. Rasch
itemfit.fit1.CCB_table2 <- itemfit(ppr1)

# check min and max infit and outfit
min(itemfit.fit1.CCB_table2$i.infitMSQ)
```

```
## [1] 0.6377949
```

```
max(itemfit.fit1.CCB_table2$i.infitMSQ)
```

```
## [1] 1.202441
```

```
min(itemfit.fit1.CCB_table2$i.outfitMSQ)
```

```
## [1] 0.6187685
```

```
max(itemfit.fit1.CCB_table2$i.outfitMSQ)
```

```
## [1] 1.214213
```

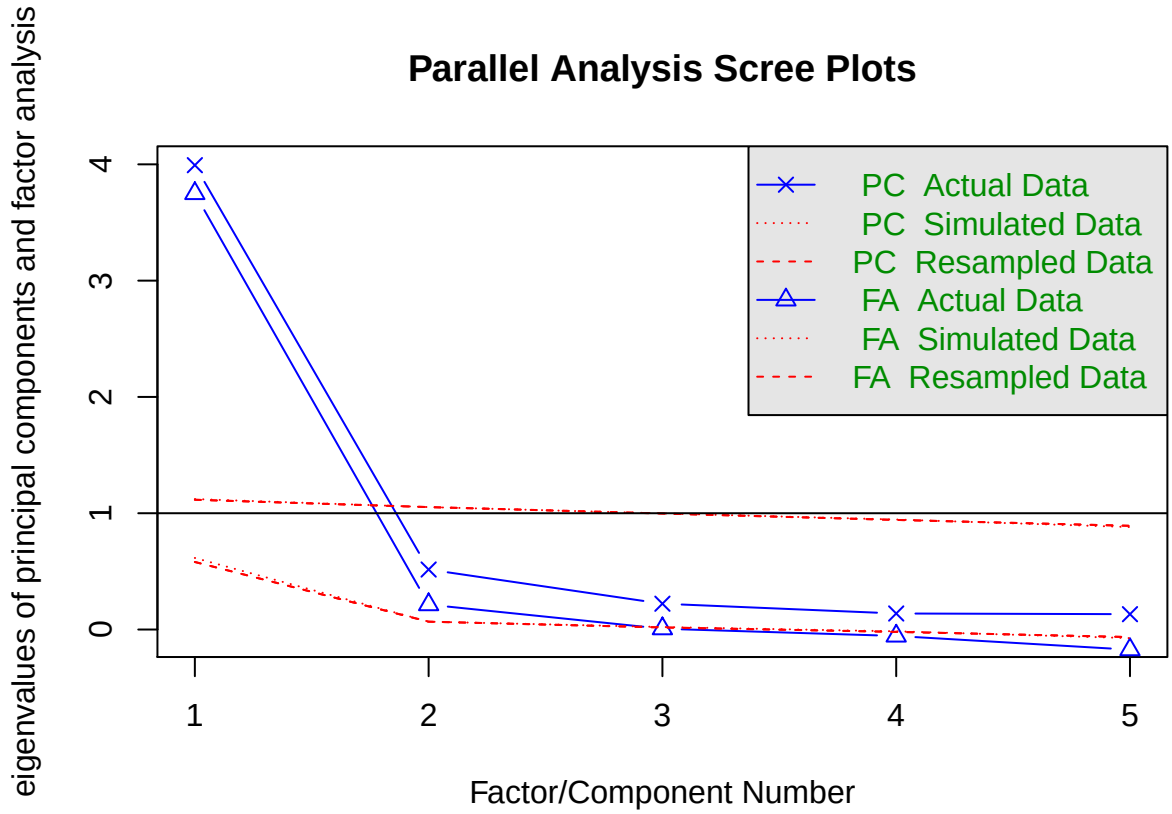
```
###################### STEP 4 - Explanatory and Confirmatory factor analysis ######################
### Exploratory factor analysis (EFA)
##correlation adequacy Bartlett's test
cortest.bartlett(CCB_table, n = nrow(CCB_table))
```

```
## R was not square, finding R from data

## $chisq
## [1] 2522.337
##
## $p.value
## [1] 0
##
## $df
## [1] 10
```

```
##sampling adequacy KMO test
KMO(CCB_table)
```

```
## Kaiser-Meyer-Olkin factor adequacy
## Call: KMO(r = CCB_table)
## Overall MSA =  0.85
## MSA for each item =
## ccb_1 ccb_2 ccb_3 ccb_4 ccb_5
##  0.86  0.83  0.89  0.82  0.83
```

```
##how many factors?
#Parallel analysis
p1 <- fa.parallel(CCB_table, fm="ml", fa="both") #both principal components and principal factors
```




```
## Parallel analysis suggests that the number of factors =  2  and the number of components =  1
```

```
png(filename="figure3.png", type="cairo", height = 6, width = 6, units = 'in', res=500)
plot(p1)
dev.off()
```

```
## pdf
##   2
```

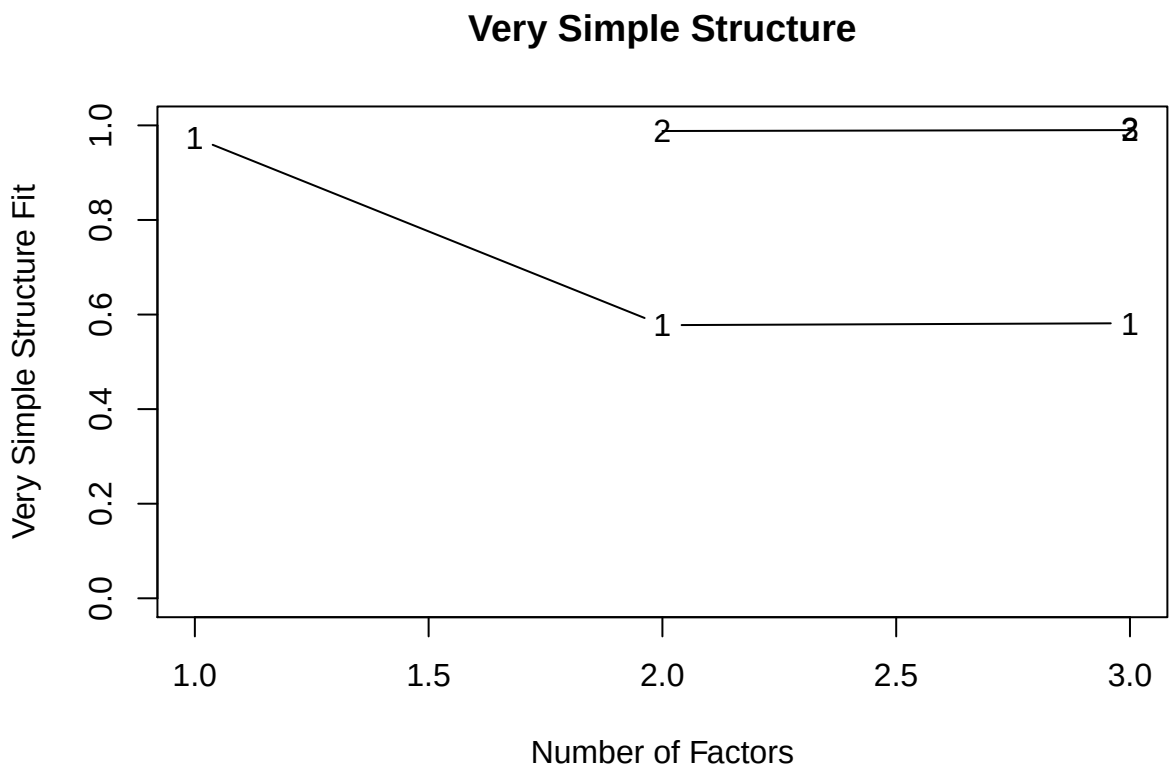
```
sum(p1$fa.values > 1.0) ##old kaiser criterion
```

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

```
sum(p1$fa.values > .7) ##new kaiser criterion
```

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

```
# very simple structure analysis
vss(CCB_table, 3)
```




```
##
## Very Simple Structure
## Call: vss(x = CCB_table, n = 3)
## VSS complexity 1 achieves a maximimum of 0.97  with  1  factors
## VSS complexity 2 achieves a maximimum of 0.99  with  3  factors
##
## The Velicer MAP achieves a minimum of 0.15  with  1  factors
## BIC achieves a minimum of  -6.05  with  2  factors
```

```
## Sample Size adjusted BIC achieves a minimum of  -2.88  with  2  factors
##
## Statistics by number of factors
##   vss1 vss2  map dof   chisq    prob sqresid  fit RMSEA   BIC SABIC complex
## 1 0.97 0.00 0.15   5 2.7e+02 1.7e-56    0.42 0.97  0.32 239.6 255.5     1.0
## 2 0.58 0.99 0.19   1 2.2e-01 6.4e-01    0.19 0.99  0.00  -6.1  -2.9     1.6
## 3 0.58 0.99 0.38  -2 2.7e-08      NA    0.14 0.99    NA    NA    NA     1.6
##    eChisq    SRMR  eCRMS eBIC
## 1 4.0e+01 6.2e-02 0.0870  8.7
## 2 1.4e-02 1.1e-03 0.0036 -6.3
## 3 1.6e-09 3.9e-07     NA   NA
```

```
# default FA - 1 factor, min residual & principal axis
fa(CCB_table,  nfactors=1, fm="minres", n.iter=10)
```

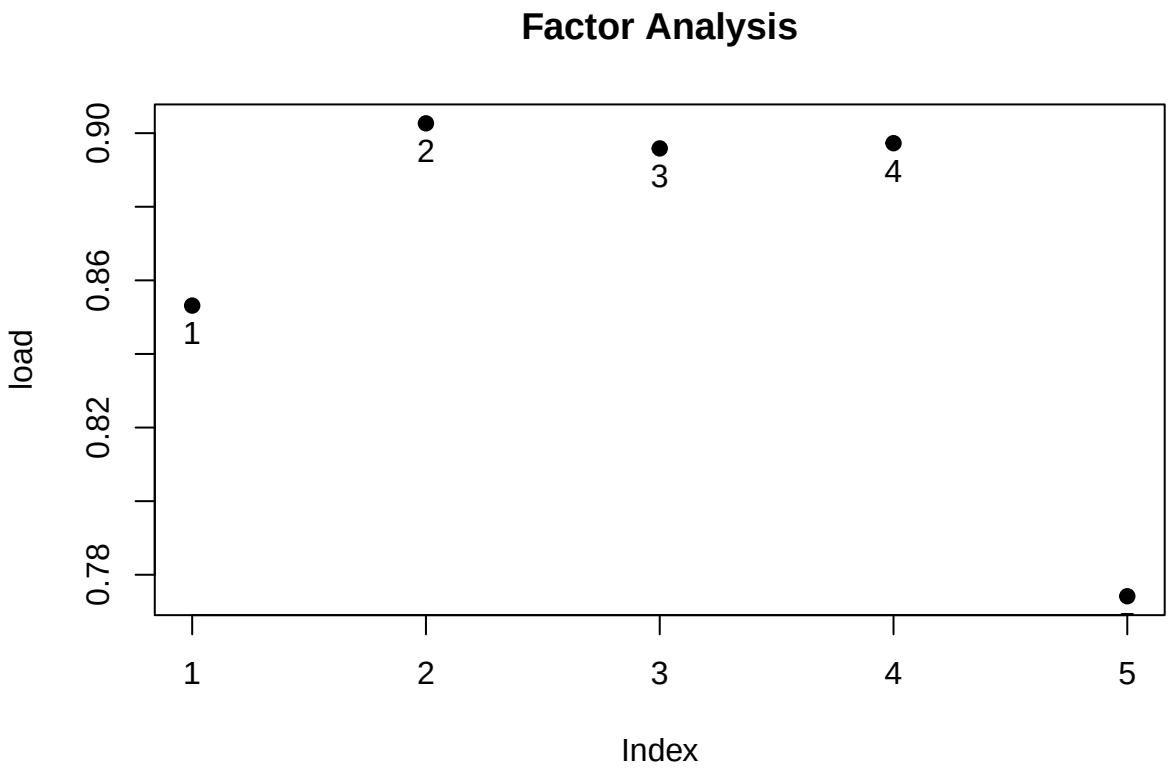
```
## Factor Analysis with confidence intervals using method = fa(r = CCB_table, nfactors = 1, n.iter = 10
## Factor Analysis using method =  minres
## Call: fa(r = CCB_table, nfactors = 1, n.iter = 10, fm = "minres")
## Standardized loadings (pattern matrix) based upon correlation matrix
##        MR1   h2   u2 com
## ccb_1 0.85 0.73 0.27   1
## ccb_2 0.90 0.82 0.18   1
## ccb_3 0.90 0.81 0.19   1
## ccb_4 0.90 0.81 0.19   1
## ccb_5 0.77 0.60 0.40   1
##
##                 MR1
## SS loadings    3.75
## Proportion Var 0.75
##
## Mean item complexity =  1
## Test of the hypothesis that 1 factor is sufficient.
##
## The degrees of freedom for the null model are  10  and the objective function was  4.79 with Chi Squa
## The degrees of freedom for the model are 5  and the objective function was  0.52
##
## The root mean square of the residuals (RMSR) is  0.06
## The df corrected root mean square of the residuals is  0.09
##
## The harmonic number of observations is  530 with the empirical chi square  40.1  with prob <  1.4e-07
## The total number of observations was  530  with Likelihood Chi Square =  270.97  with prob <  1.7e-50
##
## Tucker Lewis Index of factoring reliability =  0.788
## RMSEA index =  0.317  and the 90 % confidence intervals are  0.286 0.35
## BIC =  239.6
## Fit based upon off diagonal values = 0.99
## Measures of factor score adequacy
##                                                 MR1
## Correlation of (regression) scores with factors   0.97
## Multiple R square of scores with factors          0.95
## Minimum correlation of possible factor scores     0.89
##
##  Coefficients and bootstrapped confidence intervals
##        low  MR1 upper
```

```
## ccb_1 0.82 0.85  0.89
## ccb_2 0.87 0.90  0.92
## ccb_3 0.86 0.90  0.91
## ccb_4 0.86 0.90  0.92
## ccb_5 0.73 0.77  0.81
```

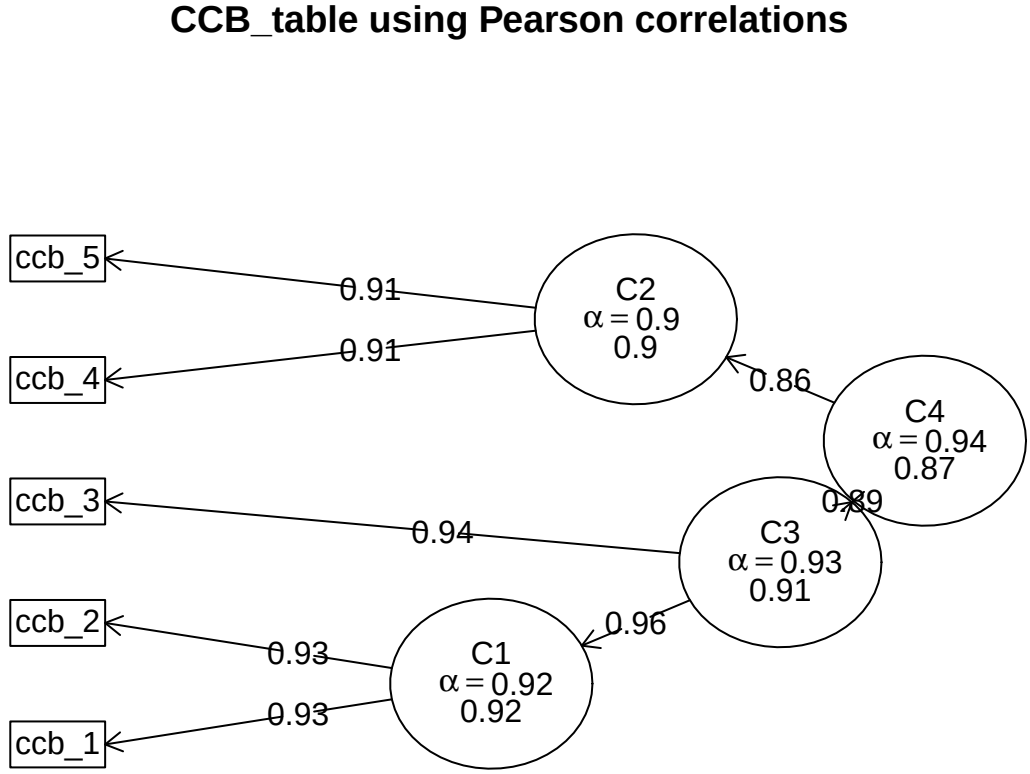
```
fa(CCB_table,  nfactors=1, fm="pa")
```

```
## Factor Analysis using method =  pa
## Call: fa(r = CCB_table, nfactors = 1, fm = "pa")
## Standardized loadings (pattern matrix) based upon correlation matrix
##        PA1   h2   u2 com
## ccb_1 0.85 0.73 0.27   1
## ccb_2 0.90 0.81 0.19   1
## ccb_3 0.90 0.80 0.20   1
## ccb_4 0.90 0.81 0.19   1
## ccb_5 0.77 0.60 0.40   1
##
##                 PA1
## SS loadings    3.75
## Proportion Var 0.75
##
## Mean item complexity =  1
## Test of the hypothesis that 1 factor is sufficient.
##
## The degrees of freedom for the null model are  10  and the objective function was  4.79 with Chi Squa
## The degrees of freedom for the model are 5  and the objective function was  0.52
##
## The root mean square of the residuals (RMSR) is  0.06
## The df corrected root mean square of the residuals is  0.09
##
## The harmonic number of observations is  530 with the empirical chi square  40.12  with prob <  1.4e-0
## The total number of observations was  530  with Likelihood Chi Square =  271.14  with prob <  1.6e-50
##
## Tucker Lewis Index of factoring reliability =  0.788
## RMSEA index =  0.317  and the 90 % confidence intervals are  0.286 0.35
## BIC =  239.77
## Fit based upon off diagonal values = 0.99
## Measures of factor score adequacy
##                                                 PA1
## Correlation of (regression) scores with factors   0.97
## Multiple R square of scores with factors          0.95
## Minimum correlation of possible factor scores     0.89
```

```
# plot the fa solution
plot(fa(CCB_table,  nfactors=1, fm="pa"))
```




```
# hierarchical cluster analysis using ICLUST (groups items)
iclust(CCB_table, title="CCB_table using Pearson correlations")
```

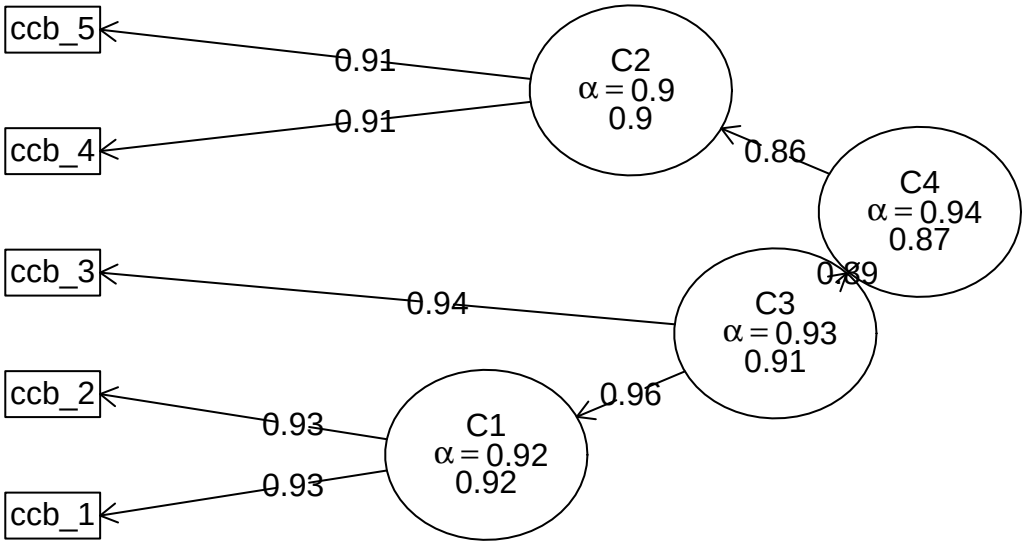



```
## ICLUST (Item Cluster Analysis)
```

```
## Call: iclust(r.mat = CCB_table, title = "CCB_table using Pearson correlations")
##
## Purified Alpha:
## [1] 0.94
##
## G6* reliability:
## [1] 1
##
## Original Beta:
## [1] 0.87
##
## Cluster size:
## [1] 5
##
## Item by Cluster Structure matrix:
##       [,1]
## ccb_1 0.85
## ccb_2 0.90
## ccb_3 0.89
## ccb_4 0.90
## ccb_5 0.79
##
## With eigenvalues of:
## [1] 3.8
##
## Purified scale intercorrelations
##  reliabilities on diagonal
##  correlations corrected for attenuation above diagonal:
##      [,1]
## [1,] 0.94
##
## Cluster fit =  0.97   Pattern fit =  1  RMSR =  0.06
```

```
summary(iclust(CCB_table))
```

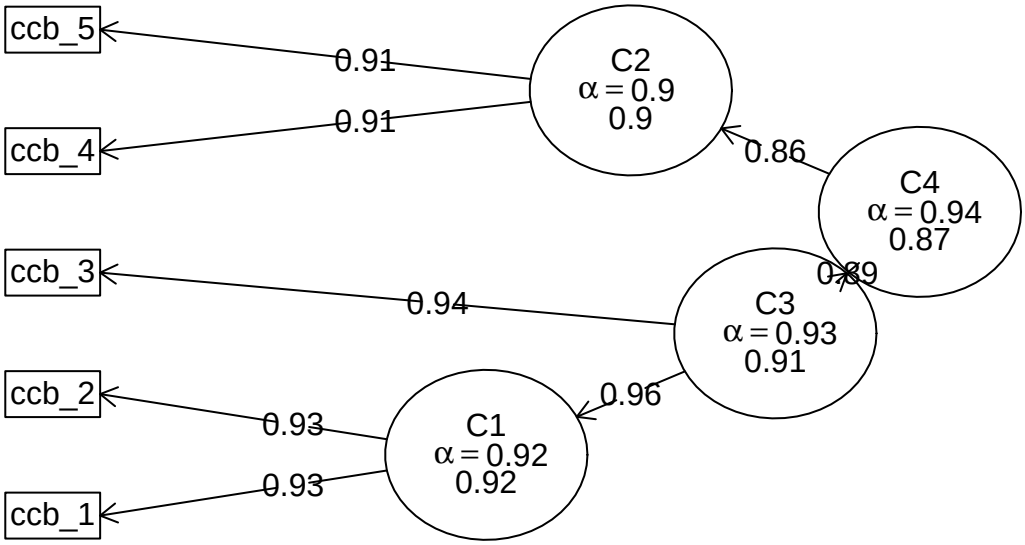
ICLUST




```
## ICLUST (Item Cluster Analysis)Call: iclust(r.mat = CCB_table)
## ICLUST
##
## Purified Alpha:
## [1] 0.94
##
##  Guttman Lambda6*
## [1] 0.94
##
## Original Beta:
## [1] 0.87
##
## Cluster size:
## [1] 5
##
## Purified scale intercorrelations
##  reliabilities on diagonal
##  correlations corrected for attenuation above diagonal:
##      [,1]
## [1,] 0.94
```

```
iclust.diagram(iclust(CCB_table, title="CCB_table using Pearson correlations"))
```

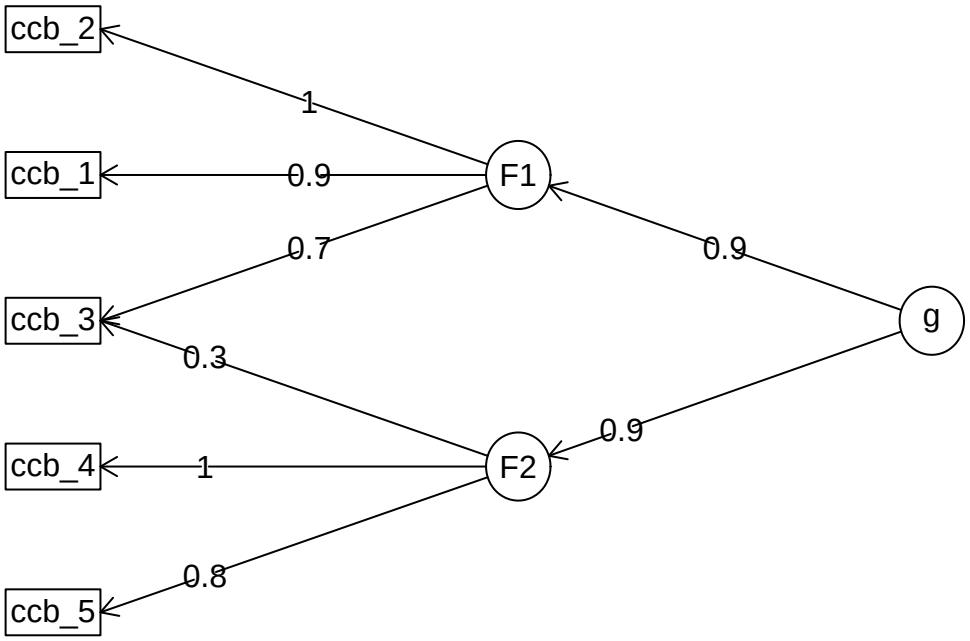
CCB_table using Pearson correlations




```
# hierarchical factor solution to find omega coefficient
omega(CCB_table, nfactors=2, sl=FALSE)
```

```
##
## Three factors are required for identification -- general factor loadings set to be equal.
## Proceed with caution.
## Think about redoing the analysis with alternative values of the 'option' setting.
```

Omega



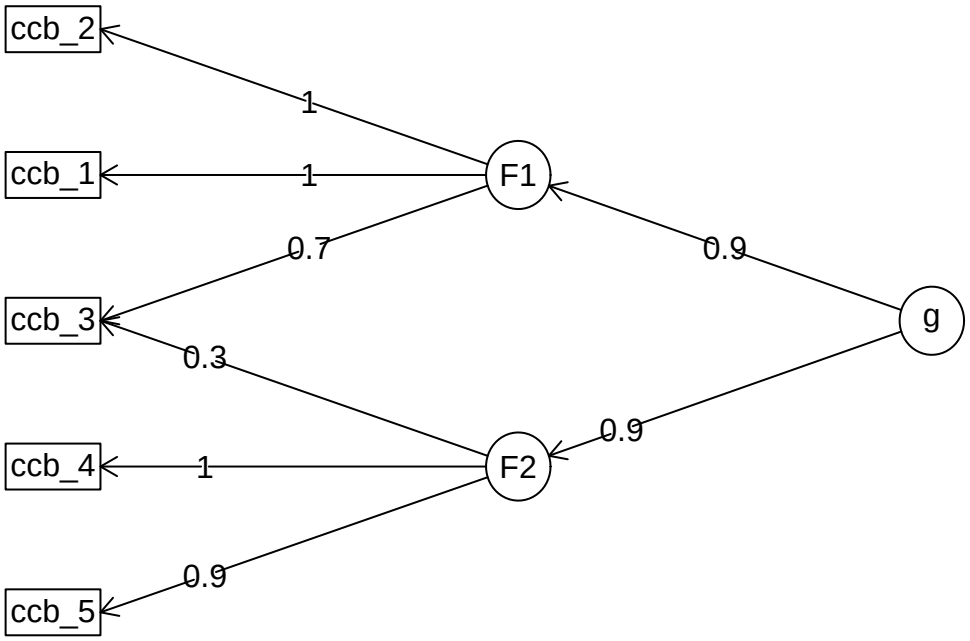

```
## Omega
## Call: omegah(m = m, nfactors = nfactors, fm = fm, key = key, flip = flip,
##     digits = digits, title = title, sl = sl, labels = labels,
##     plot = plot, n.obs = n.obs, rotate = rotate, Phi = Phi, option = option,
##     covar = covar)
## Alpha:                 0.94
## G.6:                   0.94
## Omega Hierarchical:    0.86
## Omega H asymptotic:    0.89
## Omega Total            0.96
##
## Schmid Leiman Factor loadings greater than  0.2
##          g   F1*   F2*   h2   u2   p2
## ccb_1 0.80  0.40       0.80 0.20 0.80
## ccb_2 0.85  0.43       0.92 0.08 0.80
## ccb_3 0.83  0.29       0.79 0.21 0.88
## ccb_4 0.90        0.44 1.00 0.00 0.81
## ccb_5 0.75        0.35 0.68 0.32 0.82
##
## With eigenvalues of:
##    g  F1*  F2*
## 3.42 0.43 0.33
##
## general/max  7.99   max/min =   1.3
## mean percent general =  0.82    with sd =  0.03 and cv of  0.04
## Explained Common Variance of the general factor =  0.82
##
```

```
## The degrees of freedom are 1  and the fit is  0
## The number of observations was  530  with Chi Square =  0.22  with prob <  0.64
## The root mean square of the residuals is  0
## The df corrected root mean square of the residuals is  0
## RMSEA index =  0  and the 10 % confidence intervals are  0 0.09
## BIC =  -6.05
##
## Compare this with the adequacy of just a general factor and no group factors
## The degrees of freedom for just the general factor are 5  and the fit is  0.68
## The number of observations was  530  with Chi Square =  355.12  with prob <  1.4e-74
## The root mean square of the residuals is  0.09
## The df corrected root mean square of the residuals is  0.13
##
## RMSEA index =  0.363  and the 10 % confidence intervals are  0.332 0.396
## BIC =  323.75
##
## Measures of factor score adequacy
##                                                    g   F1*  F2*
## Correlation of scores with factors             0.94  0.69 0.71
## Multiple R square of scores with factors       0.88  0.47 0.50
## Minimum correlation of factor score estimates 0.76 -0.05 0.00
##
##  Total, General and Subset omega for each subset
##                                                    g  F1*  F2*
## Omega total for total scores and subscales    0.96 0.94 0.91
## Omega general for total scores and subscales  0.86 0.78 0.74
## Omega group for total scores and subscales    0.09 0.16 0.17
```

```
# omega with polychoric matrix
CCB_table.poly <- polychoric(CCB_table)
omega(CCB_table.poly$rho, nfactors=2,  sl=FALSE)
```

```
##
## Three factors are required for identification -- general factor loadings set to be equal.
## Proceed with caution.
## Think about redoing the analysis with alternative values of the 'option' setting.
```

Omega




```
## Omega
## Call: omegah(m = m, nfactors = nfactors, fm = fm, key = key, flip = flip,
##     digits = digits, title = title, sl = sl, labels = labels,
##     plot = plot, n.obs = n.obs, rotate = rotate, Phi = Phi, option = option,
##     covar = covar)
## Alpha:                 0.95
## G.6:                   0.96
## Omega Hierarchical:    0.88
## Omega H asymptotic:    0.91
## Omega Total            0.97
##
## Schmid Leiman Factor loadings greater than  0.2
##          g   F1*   F2*   h2   u2   p2
## ccb_1 0.85  0.40       0.87 0.13 0.82
## ccb_2 0.89  0.40       0.95 0.05 0.83
## ccb_3 0.87  0.27       0.84 0.16 0.90
## ccb_4 0.92        0.39 1.00 0.00 0.84
## ccb_5 0.80        0.36 0.78 0.22 0.83
##
## With eigenvalues of:
##    g  F1*  F2*
## 3.75 0.39 0.30
##
## general/max  9.72   max/min =   1.28
## mean percent general =  0.84    with sd =  0.03 and cv of  0.04
## Explained Common Variance of the general factor =  0.85
##
```

```
## The degrees of freedom are 1  and the fit is  0
##
## The root mean square of the residuals is  0
## The df corrected root mean square of the residuals is  0.01
##
## Compare this with the adequacy of just a general factor and no group factors
## The degrees of freedom for just the general factor are 5  and the fit is  0.99
##
## The root mean square of the residuals is  0.08
## The df corrected root mean square of the residuals is  0.12
##
## Measures of factor score adequacy
##                                                  g   F1*   F2*
## Correlation of scores with factors            0.95  0.69  0.71
## Multiple R square of scores with factors      0.90  0.48  0.50
## Minimum correlation of factor score estimates 0.80 -0.04 -0.01
##
##  Total, General and Subset omega for each subset
##                                                  g  F1*  F2*
## Omega total for total scores and subscales    0.97 0.96 0.94
## Omega general for total scores and subscales  0.88 0.82 0.79
## Omega group for total scores and subscales    0.08 0.14 0.15
```

```
## Descriptive  statistics of CCB factor
attach(mydata)
```

```
## The following object is masked _by_ .GlobalEnv:
##
##     CCB_total
```

```
factor1 <- c("ccb_1", "ccb_2", "ccb_3", "ccb_4", "ccb_5")
mydata$factor1 = apply(mydata[ , factor1], 1, mean) ##creates average scores
names(mydata)
```

```
##  [1] "Age"             "Age_Categorical" "Gender"          "Education"
##  [5] "Sector"          "ccb_1"           "ccb_2"           "ccb_3"
##  [9] "ccb_4"           "ccb_5"           "CCB_total"       "factor1"
```

```
summary(mydata)
```

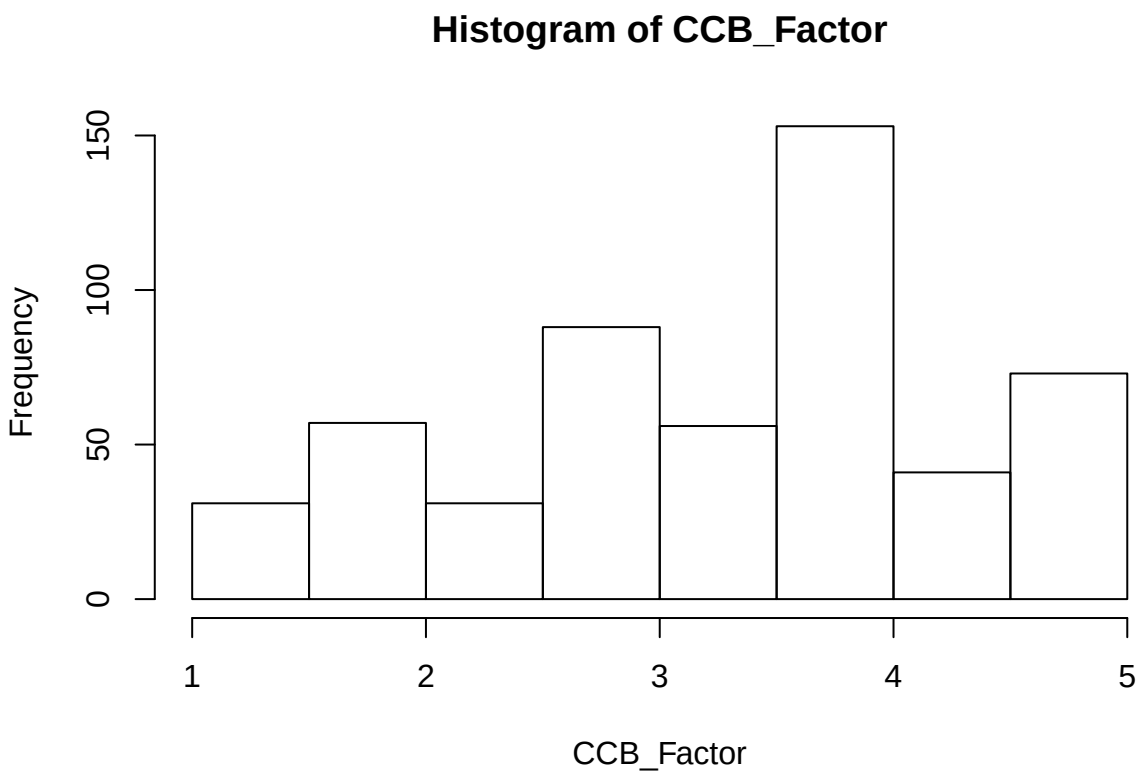
```
##       Age        Age_Categorical    Gender           Education
##  Min.   :21.00   Min.   :1.000   Length:530         Length:530
##  1st Qu.:26.00   1st Qu.:2.000   Class :character   Class :character
##  Median :30.50   Median :2.500   Mode  :character   Mode  :character
##  Mean   :31.66   Mean   :2.742
##  3rd Qu.:36.00   3rd Qu.:4.000
##  Max.   :47.00   Max.   :5.000
##      Sector          ccb_1          ccb_2           ccb_3           ccb_4
##  Min.   :1.000   Min.   :1.0   Min.   :1.000   Min.   :1.000   Min.   :1.000
##  1st Qu.:1.000   1st Qu.:3.0   1st Qu.:3.000   1st Qu.:3.000   1st Qu.:2.000
##  Median :1.000   Median :4.0   Median :4.000   Median :4.000   Median :3.000
##  Mean   :1.155   Mean   :3.4   Mean   :3.494   Mean   :3.519   Mean   :3.268
```

```
##  3rd Qu.:1.000   3rd Qu.:4.0   3rd Qu.:4.000   3rd Qu.:4.000   3rd Qu.:4.000
##  Max.   :2.000   Max.   :5.0   Max.   :5.000   Max.   :5.000   Max.   :5.000
##      ccb_5         CCB_total        factor1
##  Min.   :1.000   Min.   : 5.00   Min.   :1.000
##  1st Qu.:2.000   1st Qu.:13.00   1st Qu.:2.600
##  Median :3.000   Median :18.00   Median :3.600
##  Mean   :3.055   Mean   :16.74   Mean   :3.347
##  3rd Qu.:4.000   3rd Qu.:20.00   3rd Qu.:4.000
##  Max.   :5.000   Max.   :25.00   Max.   :5.000
```

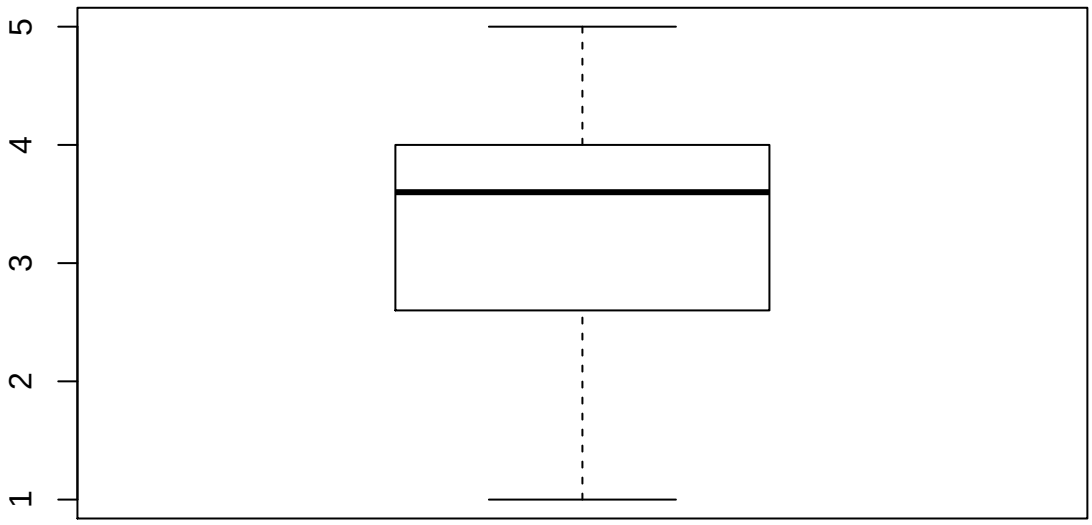
```
sd(mydata$factor1)
```

```
## [1] 1.053037
```

```
CCB_Factor <- mydata$factor1
hist(CCB_Factor)
```




```
boxplot(CCB_Factor)
```




```
?hist
```

```
## Help on topic 'hist' was found in the following packages:
##
##   Package               Library
##   mi                    /Library/Frameworks/R.framework/Versions/3.6/Resources/library
##   graphics              /Library/Frameworks/R.framework/Versions/3.6/Resources/library
##
##
## Using the first match ...
```

```
### Confirmatory factor analysis (CFA)
#specify the model
#Model1
Model1 <- "CCBs =~ ccb_1 + ccb_2 + ccb_3 + ccb_4 + ccb_5"

# fit the model
fit1 <- lavaan::cfa(Model1, data=mydata)

# model summary
summary(fit1, standardized=TRUE, fit.measures = TRUE)
```

```
## lavaan 0.6-7 ended normally after 20 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of free parameters                         10
##
##   Number of observations                           530
##
## Model Test User Model:
##
##   Test statistic                               261.563
##   Degrees of freedom                                 5
##   P-value (Chi-square)                           0.000
##
```

```
## Model Test Baseline Model:
##
##   Test statistic                                2539.105
##   Degrees of freedom                                 10
##   P-value                                         0.000
##
## User Model versus Baseline Model:
##
##   Comparative Fit Index (CFI)                     0.899
##   Tucker-Lewis Index (TLI)                        0.797
##
## Loglikelihood and Information Criteria:
##
##   Loglikelihood user model (H0)              -3055.894
##   Loglikelihood unrestricted model (H1)             NA
##
##   Akaike (AIC)                                6131.789
##   Bayesian (BIC)                              6174.518
##   Sample-size adjusted Bayesian (BIC)         6142.775
##
## Root Mean Square Error of Approximation:
##
##   RMSEA                                           0.311
##   90 Percent confidence interval - lower          0.280
##   90 Percent confidence interval - upper          0.344
##   P-value RMSEA <= 0.05                           0.000
##
## Standardized Root Mean Square Residual:
##
##   SRMR                                            0.053
##
## Parameter Estimates:
##
##   Standard errors                             Standard
##   Information                                 Expected
##   Information saturated (h1) model          Structured
##
## Latent Variables:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##   CCBs =~
##     ccb_1             1.000                               0.994    0.874
##     ccb_2             1.109    0.036   30.935    0.000    1.102    0.917
##     ccb_3             1.062    0.035   29.961    0.000    1.055    0.903
##     ccb_4             0.992    0.036   27.482    0.000    0.986    0.865
##     ccb_5             0.949    0.044   21.627    0.000    0.943    0.757
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##    .ccb_1             0.304    0.023   13.106    0.000    0.304    0.235
##    .ccb_2             0.231    0.021   11.091    0.000    0.231    0.160
##    .ccb_3             0.253    0.021   11.934    0.000    0.253    0.185
##    .ccb_4             0.326    0.024   13.373    0.000    0.326    0.251
##    .ccb_5             0.664    0.044   14.995    0.000    0.664    0.427
##     CCBs              0.988    0.078   12.607    0.000    1.000    1.000
```

```r
#Modification indices
modindices(fit1, minimum.value = 10, sort = TRUE)
```

```
##      lhs op   rhs      mi    epc sepc.lv sepc.all sepc.nox
## 21 ccb_4 ~~ ccb_5 202.194  0.341   0.341    0.733    0.733
## 12 ccb_1 ~~ ccb_2 113.979  0.204   0.204    0.769    0.769
## 17 ccb_2 ~~ ccb_4  44.599 -0.128  -0.128   -0.466   -0.466
## 18 ccb_2 ~~ ccb_5  38.612 -0.142  -0.142   -0.364   -0.364
## 14 ccb_1 ~~ ccb_4  34.363 -0.109  -0.109   -0.346   -0.346
## 15 ccb_1 ~~ ccb_5  23.537 -0.114  -0.114   -0.254   -0.254
```

```r
#fit indices
fitmeasures(fit1, c("gfi", "agfi", "nfi", "cfi", "tli", "rmsea", "srmr","aic","bic"))
```

```
##      gfi     agfi      nfi      cfi      tli    rmsea     srmr      aic
##    0.833    0.500    0.897    0.899    0.797    0.311    0.053 6131.789
##      bic
## 6174.518
```

```r
#Model2
Model2 <- "CCBs =~ ccb_1 + ccb_2 + ccb_3 + ccb_4 + ccb_5
ccb_4~~ccb_5 + ccb_3"

# fit the model
fit2 <- lavaan::cfa(Model2, data=mydata)

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

```
## lavaan 0.6-7 ended normally after 30 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of free parameters                         12
##
##   Number of observations                           530
##
## Model Test User Model:
##
##   Test statistic                                20.418
##   Degrees of freedom                                 3
##   P-value (Chi-square)                           0.000
##
## Model Test Baseline Model:
##
##   Test statistic                              2539.105
##   Degrees of freedom                                10
##   P-value                                        0.000
##
## User Model versus Baseline Model:
##
##   Comparative Fit Index (CFI)                    0.993
```

```
##   Tucker-Lewis Index (TLI)                       0.977
##
## Loglikelihood and Information Criteria:
##
##   Loglikelihood user model (H0)              -2935.322
##   Loglikelihood unrestricted model (H1)             NA
##
##   Akaike (AIC)                                5894.644
##   Bayesian (BIC)                              5945.919
##   Sample-size adjusted Bayesian (BIC)         5907.827
##
## Root Mean Square Error of Approximation:
##
##   RMSEA                                          0.105
##   90 Percent confidence interval - lower         0.065
##   90 Percent confidence interval - upper         0.150
##   P-value RMSEA <= 0.05                          0.014
##
## Standardized Root Mean Square Residual:
##
##   SRMR                                           0.019
##
## Parameter Estimates:
##
##   Standard errors                             Standard
##   Information                                 Expected
##   Information saturated (h1) model          Structured
##
## Latent Variables:
##                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##   CCBs =~
##     ccb_1            1.000                               1.019    0.896
##     ccb_2            1.118    0.032   34.890    0.000    1.139    0.947
##     ccb_3            1.010    0.034   29.930    0.000    1.029    0.880
##     ccb_4            0.903    0.036   25.322    0.000    0.920    0.813
##     ccb_5            0.849    0.044   19.331    0.000    0.865    0.694
##
## Covariances:
##                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##  .ccb_4 ~~
##    .ccb_5            0.356    0.032   11.205    0.000    0.356    0.602
##  .ccb_3 ~~
##    .ccb_4            0.079    0.016    4.915    0.000    0.079    0.216
##
## Variances:
##                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##    .ccb_1            0.255    0.021   11.897    0.000    0.255    0.197
##    .ccb_2            0.148    0.020    7.448    0.000    0.148    0.102
##    .ccb_3            0.308    0.024   12.592    0.000    0.308    0.226
##    .ccb_4            0.434    0.029   14.749    0.000    0.434    0.339
##    .ccb_5            0.805    0.052   15.372    0.000    0.805    0.518
##     CCBs             1.038    0.079   13.100    0.000    1.000    1.000
```

```
#Modification indices
modindices(fit2, minimum.value = 10, sort = TRUE)
```

```
##      lhs op   rhs     mi   epc sepc.lv sepc.all sepc.nox
## 14 ccb_1 ~~ ccb_2 20.808 0.156   0.156    0.801    0.801
## 21 ccb_3 ~~ ccb_5 20.808 0.119   0.119    0.240    0.240
```

```
fitmeasures(fit2, c("gfi", "agfi", "nfi", "cfi", "tli", "rmsea", "srmr","aic","bic"))
```

```
##      gfi     agfi      nfi      cfi      tli    rmsea     srmr      aic
##    0.984    0.922    0.992    0.993    0.977    0.105    0.019 5894.644
##      bic
## 5945.919
```

```
# coefficients only
coef(fit2)
```

```
##  CCBs=~ccb_2  CCBs=~ccb_3  CCBs=~ccb_4  CCBs=~ccb_5 ccb_4~~ccb_5 ccb_3~~ccb_4
##        1.118        1.010        0.903        0.849        0.356        0.079
## ccb_1~~ccb_1 ccb_2~~ccb_2 ccb_3~~ccb_3 ccb_4~~ccb_4 ccb_5~~ccb_5   CCBs~~CCBs
##        0.255        0.148        0.308        0.434        0.805        1.038
```

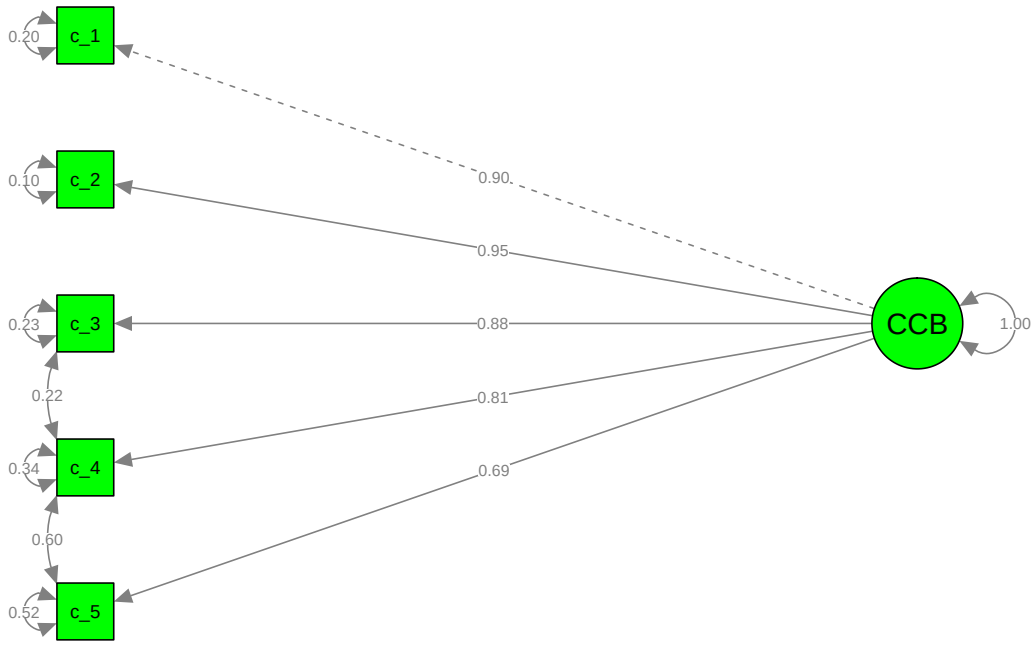
```
# R square
inspect(fit2, 'r2')
```

```
## ccb_1 ccb_2 ccb_3 ccb_4 ccb_5
## 0.803 0.898 0.774 0.661 0.482
```

```
fitmeasures(fit2)
```

```
##                npar                fmin               chisq
##              12.000               0.019              20.418
##                  df              pvalue      baseline.chisq
##               3.000               0.000            2539.105
##         baseline.df     baseline.pvalue                 cfi
##              10.000               0.000               0.993
##                 tli                nnfi                 rfi
##               0.977               0.977               0.973
##                 nfi                pnfi                 ifi
##               0.992               0.298               0.993
##                 rni                logl   unrestricted.logl
##               0.993           -2935.322                  NA
##                 aic                 bic              ntotal
##            5894.644            5945.919             530.000
##                bic2               rmsea      rmsea.ci.lower
##            5907.827               0.105               0.065
##      rmsea.ci.upper        rmsea.pvalue                 rmr
##               0.150               0.014               0.026
##          rmr_nomean                srmr        srmr_bentler
##               0.026               0.019               0.019
## srmr_bentler_nomean                crmr         crmr_nomean
##               0.019               0.022               0.022
##           srmr_mplus    srmr_mplus_nomean             cn_05
##               0.018               0.018             203.853
##               cn_01                 gfi                agfi
##             295.488               0.984               0.922
##                pgfi                 mfi                ecvi
##               0.197               0.984               0.084
```

```
# CFA diagram from psych package
semPaths(fit2,what="paths",whatLabels="stand", layout = "tree", color = "green", rotation = 4)
```




```
###################### STEP 5 - CCT ######################
# CTT for a single scale

# Alpha by bootstrapping
ci.reliability(data=CCB_table, type="alpha", conf.level = 0.95, interval.type="perc", B=100)
```

```
## $est
## [1] 0.9357009
##
## $se
## [1] 0.005472618
##
## $ci.lower
## [1] 0.9216912
##
## $ci.upper
## [1] 0.9462157
##
## $conf.level
## [1] 0.95
##
## $type
## [1] "alpha"
##
## $interval.type
## [1] "percentile bootstrap"
```

```r
# Guttman lambda 6 (G6) and Beta values
splitHalf(CCB_table)
```

```
## Split half reliabilities
## Call: splitHalf(r = CCB_table)
##
## Maximum split half reliability (lambda 4) =  0.93
## Guttman lambda 6                          =  0.94
## Average split half reliability            =  0.9
## Guttman lambda 3 (alpha)                  =  0.94
## Guttman lambda 2                          =  0.94
## Minimum split half reliability  (beta)    =  0.87
## Average interitem r =  0.75  with median =  0.77
```

```r
# Omega
ci.reliability(data=CCB_table, type="omega", conf.level = 0.95, interval.type="perc", B=100)
```

```
## $est
## [1] 0.9355858
##
## $se
## [1] 0.005180341
##
## $ci.lower
## [1] 0.9255562
##
## $ci.upper
## [1] 0.9451029
##
## $conf.level
## [1] 0.95
##
## $type
## [1] "omega"
##
## $interval.type
## [1] "percentile bootstrap"
```

```r
###################### STEP 6 ####################
# check everything about your scores

# check descriptives
summary(CCB_table)
```

```
##      ccb_1          ccb_2           ccb_3           ccb_4           ccb_5
##  Min.   :1.0   Min.   :1.000   Min.   :1.000   Min.   :1.000   Min.   :1.000
##  1st Qu.:3.0   1st Qu.:3.000   1st Qu.:3.000   1st Qu.:2.000   1st Qu.:2.000
##  Median :4.0   Median :4.000   Median :4.000   Median :3.000   Median :3.000
##  Mean   :3.4   Mean   :3.494   Mean   :3.519   Mean   :3.268   Mean   :3.055
##  3rd Qu.:4.0   3rd Qu.:4.000   3rd Qu.:4.000   3rd Qu.:4.000   3rd Qu.:4.000
##  Max.   :5.0   Max.   :5.000   Max.   :5.000   Max.   :5.000   Max.   :5.000
```

```r
# Histograms
png(filename="figure1.png", type="cairo", height = 6, width = 6, units = 'in', res=300)
hist(CCB_total, breaks=40 , border=F , col=rgb(0.1,0.8,0.3,0.5) , xlab="distribution of WEMWBS_total" ,
dev.off()
```

```
## pdf
##   2
```

```r
# Plot png image
png(filename="figure2.png", type="cairo", height = 8, width = 8, units = 'in', res=300)
cor.plot(lowerCor(CCB_table, method = "spearman"), numbers=TRUE, main="Correlations between CCB items",
         cex=0.5, cex.axis=0.7, xlas = 2)
```

```
##       ccb_1 ccb_2 ccb_3 ccb_4 ccb_5
## ccb_1 1.00
## ccb_2 0.85  1.00
## ccb_3 0.77  0.81  1.00
## ccb_4 0.70  0.74  0.77  1.00
## ccb_5 0.60  0.63  0.65  0.83  1.00
```

```r
dev.off()
```

```
## pdf
##   2
```