efa\_to\_run.RMD

# First we set the analysis constants

# rotation method for EFA factors  
rotate = "varimax"  
# filtering threshold for filtering questions / items in EFA  
loading\_thresh = 0.4

# Filter data, remove ids with too many missings, also remove outliers

set.seed(2124)  
res = filter\_multi\_df\_outliers\_missing(  
 df=df,  
 value\_cols=q\_cols)

##   
## iter imp variable  
## 1 1  
## 1 2  
## 1 3  
## 1 4  
## 1 5  
## 2 1  
## 2 2  
## 2 3  
## 2 4  
## 2 5  
## 3 1  
## 3 2  
## 3 3  
## 3 4  
## 3 5  
## 4 1  
## 4 2  
## 4 3  
## 4 4  
## 4 5  
## 5 1  
## 5 2  
## 5 3  
## 5 4  
## 5 5

df = res[["df"]]  
res[["missing\_df"]]

## [1] id q7 q8 q9 q10 q11 q13 q15 q16 q17 q18 q19 q20 q22 q23 q24 q25 q26 q27  
## [20] q28 q29 q30 q31 q32 q33 q35 q36 q37 q39 q40  
## <0 rows> (or 0-length row.names)

res[["outlier\_df"]][["id"]]

## [1] "patient12" "patient15" "patient17" "patient18" "patient21"   
## [6] "patient51" "patient59" "patient84" "patient93" "patient94"   
## [11] "patient95" "patient105" "patient127" "patient129" "patient151"  
## [16] "patient155" "patient162" "patient167" "patient171" "patient196"

# Check EFA assumptions

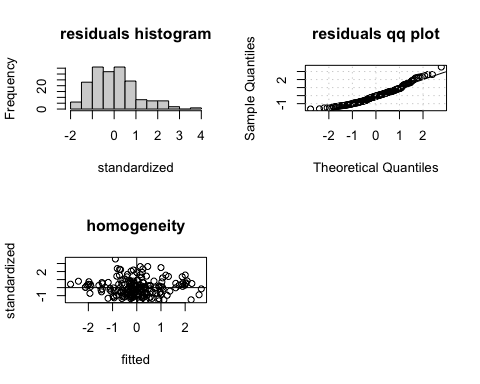
res = check\_efa\_assumptions(  
 df=df,  
 value\_cols=q\_cols,  
 figs\_path=NULL)  
# bartlet correlation test  
res[["barlet\_test"]][["p.value"]]

## [1] 0

# kaiser sampling adequecy  
res[["kaiser\_sampling\_adequacy"]]

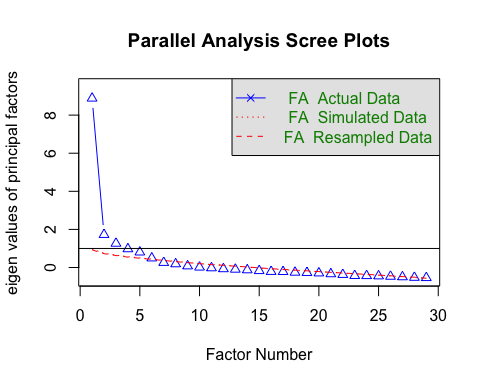
## Kaiser-Meyer-Olkin factor adequacy  
## Call: psych::KMO(r = cor\_mat)  
## Overall MSA = 0.88  
## MSA for each item =   
## q7 q8 q9 q10 q11 q13 q15 q16 q17 q18 q19 q20 q22 q23 q24 q25   
## 0.88 0.91 0.86 0.84 0.87 0.88 0.91 0.90 0.91 0.79 0.78 0.86 0.91 0.84 0.93 0.81   
## q26 q27 q28 q29 q30 q31 q32 q33 q35 q36 q37 q39 q40   
## 0.88 0.90 0.82 0.86 0.90 0.90 0.93 0.90 0.91 0.91 0.92 0.85 0.84

res[["plt\_func"]]()



# Get number of factors using parallel analysis

res = get\_factor\_num(  
 df=df,  
 value\_cols=q\_cols,  
 figs\_path=NULL)



## Parallel analysis suggests that the number of factors = 6 and the number of components = NA

# R returns: Number of factors = with eigen values > eigen values of random data  
factor\_num = res[["factor\_num"]]  
factor\_num

## [1] 6

# Fit the EFA with those number of factors and remove items

res = fit\_fa\_model(  
 df=df,  
 factor\_num=factor\_num,  
 rotate=rotate,  
 value\_cols=q\_cols,  
 loading\_thresh=loading\_thresh)  
  
df\_filtered = res[["df\_filtered"]]  
# Tucker Lewis Index  
res[["tuker\_lewis\_index"]]

## [1] 0.9219351

# RMSEA index  
res[["rmsea"]]

## RMSEA lower upper confidence   
## 0.04859302 0.03708448 0.06011440 0.90000000

res[["rejected\_items"]]

## [1] "q15" "q17" "q19"

res[["cfi"]]

## [1] 0.9538159

accepted\_items = res[["accepted\_items"]]

# Fit the EFA using filtered data  
res = fit\_fa\_model(  
 df=df\_filtered,  
 factor\_num=factor\_num,  
 value\_cols=accepted\_items,  
 rotate=rotate,  
 fm="ml",  
 loading\_thresh=loading\_thresh)  
  
# Tucker Lewis Index  
res[["tuker\_lewis\_index"]]

## [1] 0.932205

# RMSEA index  
res[["rmsea"]]

## RMSEA lower upper confidence   
## 0.04862791 0.03508636 0.06185560 0.90000000

# Checking if any item is rejected this time  
res[["rejected\_items"]]

## character(0)

res[["cfi"]]

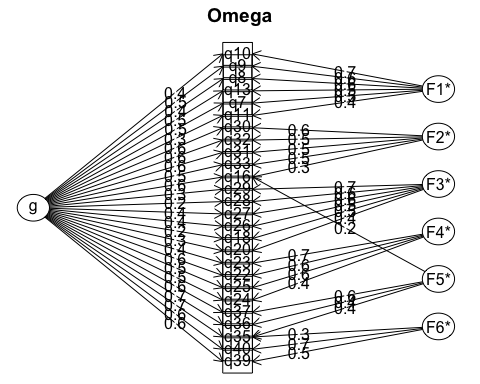
## [1] 0.9626483

res[["factor\_item\_list"]]

## [[1]]  
## [1] "q7" "q8" "q9" "q10" "q11" "q13"  
##   
## [[2]]  
## [1] "q16" "q30" "q31" "q32" "q33"  
##   
## [[3]]  
## [1] "q18" "q20" "q26" "q27" "q28" "q29"  
##   
## [[4]]  
## [1] "q22" "q23" "q24" "q25"  
##   
## [[5]]  
## [1] "q35" "q36" "q37"  
##   
## [[6]]  
## [1] "q39" "q40"

# Calculate MacDonalds’ Omega overall and for accepted items

omega\_model = omega(m=df[ , accepted\_items], nfactors=factor\_num)



# The ω\_h coefficient  
omega\_model[["omega\_h"]]

## [1] 0.6862047

# limit of omega  
omega\_model[["omega.lim"]]

## [1] 0.7234134

# Cronbach Alpha  
omega\_model[["alpha"]]

## [1] 0.9161527

# The omega\_t coefficient  
omega\_model[["omega.tot"]]

## [1] 0.9485651

# The summary statistics for the omega total, omega hierarchical (general) and omega within each group.  
omega\_model[["omega.group"]]

## total general group  
## g 0.9485651 0.6862047 0.1628795  
## F1\* 0.8656403 0.3179127 0.5477276  
## F2\* 0.8557106 0.5259727 0.3297379  
## F3\* 0.7814615 0.1692158 0.6122457  
## F4\* 0.8386665 0.3658332 0.4728332  
## F5\* 0.8133673 0.5210201 0.2923472  
## F6\* 0.8227657 0.4056678 0.4170979