Estudio de la Recuperación de un Análisis Factorial Confirmatorio sobre una matriz de varianzas y covarianzas cuyos datos presentaban valores perdidos

Alicia Gil Matute

2024-06-02

INTRODUCCIÓN

El análisis factorial confirmatorio (AFC) es la técnica estadística mediante la cual podemos establecer la estructura subyacente de una serie de variables observadas y así poder medir y usar variables no observables. Dicha técnica utiliza la matriz de varianzas y covarianzas para evaluar si los datos ajustan al modelo teórico, así que para realizar un AFC sobre unos datos, podemos introducir como input los datos en bruto o directamente la matriz de varianzas y covarianzas.

Sin embargo, si en nuestra base de datos existen valores perdidos, el software con el cual vayamos a trabajar va a utilizar alguna estrategia para poder realizar los cálculos sin estos valores perdidos (la más común es "eliminación por lista"). Esto provoca que nuestros análisis no estén utilizando todas las observaciones recogidas, sino muchas menos. Por ello existen estrategias modernas para no desechar tantos casos, como la imputación múltiple o la imputación por máxima verosimilitud.

Este trabajo pretende poner a prueba la estrategia de imputación múltiple para realizar un AFC el cual recibe como input una matriz de varianzas y covarianzas calculada sobre una base de datos. Una base de datos sobre la que se han simulado valores perdidos MCAR y se han imputado mediante dicha estrategia. Queremos estudiar como recupera los parametros estimados de un AFC y sus indices de ajuste y compararlo con el modelo AFC que recibe como input la matriz de varianzas y covarianzas sin simular valores perdidos.

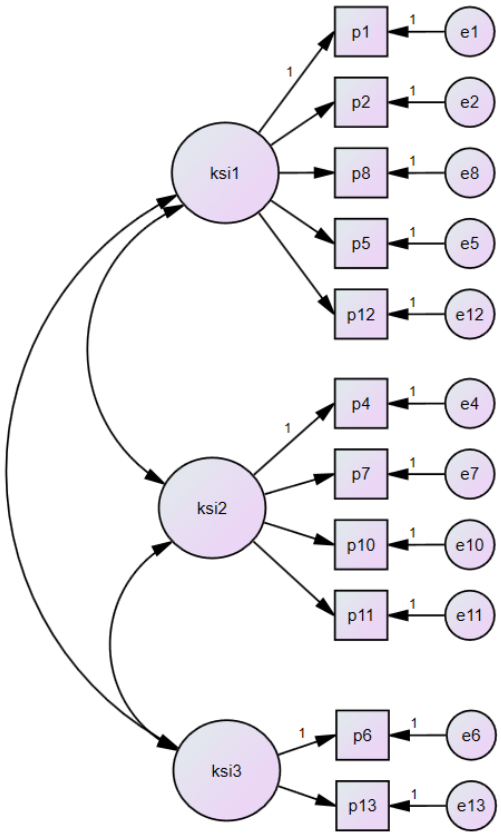
OBJETIVOS E HIPÓTESIS

El objetivo de este trabajo es el estudio de la recuperación de los parámetros estimados y los índices de ajuste de un modelo AFC que recibe como input una matriz de varianzas y covarianzas de unos datos imputados. De este manera se podrá comprobar si el método de imputación múltiple es recomendable para emplearse en modelos AFC.

Nuestra hipótesis vaticina que las estimaciones y los índices de ajuste del modelo AFC imputado son muy similares a los del modelo AFC sin imputar.

MÉTODO

La base de datos empleada es la llamada "Liderazgo.sav" donde se incluyen 15 variables de las cuales 13 son ítems que miden 3 factores relacionados con el liderazgo, pero de los cuales solo 11 forman parte de la estructura factorial (los items 3 y 9 no forman parte de la estructura factorial, como se puede ver en la imagen). Los ítems están medidos en una escala Likert con valores 1-7. El tamaño de la muestra es de 96 sujetos y todos los análisis se han realizado con el software R.



En primer lugar, se cargan los archivos y se eliminan de la base de datos los ítems 3 y 9 que no forman parte de la estructura, y 2 variables que servían a otros objetivos del estudio. Después se calcula la matriz de varianzas y covarianzas de los datos originales que utilizaremos después para extraer la matriz residual y tener una primera impresión de sí se ha realizado bien o no la imputación múltiple.

En segundo lugar, simulamos valores perdidos MCAR en el 40% de la muestra en todos los ítems y empleamos la librería mice para realizar la imputación múltiple. Se imputan 20 bases de datos mediante el método "pmm" (predictive mean matching) y después, mediante un bucle for() se realizan las matrices de varianzas y covarianzas de cada una de las 20 bases de datos imputadas.A continuación, calculamos la matriz promedio de esas 20 matrices de varianzas y covarianzas.

En tercer lugar, especificamos el modelo de AFC que se ha presentado en la imagen e introducimos como input esa matriz promedio e indicamos que el método de estimación sea máxima verosimilitud ("ML"). Indicamos también que imprima los índices de ajuste.

Por último, calculamos el modelo AFC con los datos originales (sin valores perdidos) pero introduciendo también como input la matriz de varianzas y covarianzas.

RESULTADOS

Índices de Ajuste AFC imputado VS Índices de Ajuste AFC original

```
### Usando `gridExtra` en RMarkdown
```

```r
library(gridExtra)
library(png)
library(grid)

# Leer las imágenes
img1 <- readPNG("ajuste_imp.png")
img2 <- readPNG("ajuste_raw.png")

# Convertir las imágenes a objetos grid
g1 <- rasterGrob(img1, interpolate=TRUE)
g2 <- rasterGrob(img2, interpolate=TRUE)

# Mostrar las imágenes lado a lado
grid.arrange(g1, g2, ncol=2)
```

```
lavaan 0.6.17 ended normally after 29 iterations

  Estimator                                         ML
  Optimization method                           NLMINB
  Number of model parameters                        36

  Number of observations                            96

Model Test User Model:

  Test statistic                               172.399
  Degrees of freedom                                41
  P-value (Chi-square)                           0.000

Model Test Baseline Model:

  Test statistic                              1047.698
  Degrees of freedom                                55
  P-value                                        0.000

User Model versus Baseline Model:

  Comparative Fit Index (CFI)                    0.868
  Tucker-Lewis Index (TLI)                       0.822

Loglikelihood and Information Criteria:

  Loglikelihood user model (H0)              -1582.997
  Loglikelihood unrestricted model (H1)      -1496.797

  Akaike (AIC)                                3237.993
  Bayesian (BIC)                              3330.310
  Sample-size adjusted Bayesian (SABIC)       3216.642

Root Mean Square Error of Approximation:

  RMSEA                                          0.183
  90 Percent confidence interval - lower         0.155
  90 Percent confidence interval - upper         0.211
  P-value H_0: RMSEA <= 0.050                    0.000
  P-value H_0: RMSEA >= 0.080                    1.000

Standardized Root Mean Square Residual:

  SRMR                                           0.047
```

```
lavaan 0.6.17 ended normally after 28 iterations

  Estimator                                         ML
  Optimization method                           NLMINB
  Number of model parameters                        36

  Number of observations                            96

Model Test User Model:

  Test statistic                                80.412
  Degrees of freedom                                41
  P-value (Chi-square)                           0.000

Model Test Baseline Model:

  Test statistic                               938.310
  Degrees of freedom                                55
  P-value                                        0.000

User Model versus Baseline Model:

  Comparative Fit Index (CFI)                    0.955
  Tucker-Lewis Index (TLI)                       0.940

Loglikelihood and Information Criteria:

  Loglikelihood user model (H0)              -1592.065
  Loglikelihood unrestricted model (H1)      -1551.859

  Akaike (AIC)                                3256.131
  Bayesian (BIC)                              3348.447
  Sample-size adjusted Bayesian (SABIC)       3234.780

Root Mean Square Error of Approximation:

  RMSEA                                          0.100
  90 Percent confidence interval - lower         0.067
  90 Percent confidence interval - upper         0.132
  P-value H_0: RMSEA <= 0.050                    0.009
  P-value H_0: RMSEA >= 0.080                    0.853

Standardized Root Mean Square Residual:

  SRMR                                           0.037
```

Como podemos observar, el valor del estadístico de chi-cuadrado resulta significativo en ambso modelos lo que signfica que el modelo propuesto ajusta a los datos, pero con valores de chi-cuadrado diferentes, en el modelo de AFC imputado se obtiene un valor de chi-cuadrado de 172.399 y en el modelo de AFC original un chi-cuadrado de 80.412. Igualmente, estan empleando la misma cantidad de información porque ambos están empleando 41 grados de libertad. Por otro lado, el índice de RMSEA que también evalua el ajuste del modelo a los datos resulta mayor a 0.05 en ambos modelos, con lo cual recibimos información contradictoria. Como el estadístico chi-cuadrado es tan sensible a la muestra, RMSEA resulta ser un índice más robusto con lo cual, basandonos en él, tanto el modelo AFC imputado como el modelo AFC original tiene un mal ajuste a los datos. Está cuestión, se comentará en los siguientes apartados.

Respecto a los índices TLI y CFI hay diferencias más claras entre ambos modelos. Mientras que en modelo

AFC imputado los valores estimados de TLI como CFI no superan el 0.95, en el modelo de AFC original sí son más próximos a 0.95. Con lo cual, el modelo AFC imputado recupera mal la mejora del ajuste respecto al modelo nulo.

Estimaciones de parámetros modelo AFC imputado VS estimaciones de parámetros modelo AFC original

```
# Leer las imágenes
img1 <- readPNG("estimaciones_imp.png")
img2 <- readPNG("estimaciones_raw.png")

# Convertir las imágenes a objetos grid
g1 <- rasterGrob(img1, interpolate=TRUE, width=unit(8, "cm"), height=unit(9, "cm"))
g2 <- rasterGrob(img2, interpolate=TRUE, width=unit(8, "cm"), height=unit(9, "cm"))

# Mostrar las imágenes lado a lado
grid.arrange(g1, g2, ncol=2)
```

Latent Variables:

	Estimate	Std.Err	z-value	P(>|z|)	Std.lv	Std.all
factor1 =~						
p1	1.326	0.129	10.255	0.000	1.326	0.850
p2	1.322	0.129	10.228	0.000	1.322	0.849
p8	1.166	0.128	9.104	0.000	1.166	0.786
p5	1.393	0.131	10.646	0.000	1.393	0.870
p12	1.392	0.134	10.358	0.000	1.392	0.855
factor2 =~						
p4	1.048	0.152	6.892	0.000	1.048	0.643
p7	1.140	0.119	9.579	0.000	1.140	0.816
p10	1.469	0.144	10.180	0.000	1.469	0.848
p11	1.607	0.151	10.670	0.000	1.607	0.874
factor3 =~						
p6	1.557	0.172	9.028	0.000	1.557	0.789
p13	1.639	0.136	12.063	0.000	1.639	0.958

Covariances:

	Estimate	Std.Err	z-value	P(>|z|)	Std.lv	Std.all
factor1 ~~						
factor2	0.944	0.024	39.471	0.000	0.944	0.944
factor3	0.865	0.039	22.183	0.000	0.865	0.865
factor2 ~~						
factor3	0.891	0.038	23.667	0.000	0.891	0.891

Latent Variables:

	Estimate	Std.Err	z-value	P(>|z|)	Std.lv	Std.all
factor1 =~						
p1	1.271	0.130	9.744	0.000	1.271	0.823
p2	1.445	0.137	10.527	0.000	1.445	0.864
p8	1.160	0.120	9.667	0.000	1.160	0.819
p5	1.430	0.132	10.819	0.000	1.430	0.879
p12	1.410	0.133	10.594	0.000	1.410	0.867
factor2 =~						
p4	1.009	0.157	6.439	0.000	1.009	0.611
p7	1.084	0.127	8.566	0.000	1.084	0.758
p10	1.527	0.139	10.961	0.000	1.527	0.890
p11	1.595	0.148	10.770	0.000	1.595	0.880
factor3 =~						
p6	1.439	0.168	8.561	0.000	1.439	0.763
p13	1.684	0.138	12.226	0.000	1.684	0.976

Covariances:

	Estimate	Std.Err	z-value	P(>|z|)	Std.lv	Std.all
factor1 ~~						
factor2	0.905	0.030	30.471	0.000	0.905	0.905
factor3	0.830	0.044	18.665	0.000	0.830	0.830
factor2 ~~						
factor3	0.836	0.046	18.318	0.000	0.836	0.836

CONCLUSIONES

Antes de comentar los resultados es importante aclarar lo siguiente: este trabajo no pretende evaluar el ajuste del modelo AFC original ni mejorarlo, y mucho menos pretendemos que el modelo AFC imputado supere en ajuste al modelo original, es imposible dado que solo podemos aspirar a que se parezca al ajuste original. Si

el modelo AFC original tiene un ajuste mediocre (como es el caso) el ajuste del modelo imputado, por mucho que pueda parecerse al original, será igual de mediocre. Y lo mismo ocurre con los valores estimados para los pesos factoriales, si el modelo original tiene problemas obtendremos pesos factoriales mayores a 1, también los obtendrá el modelo imputado. Así que, lo que nos interesa evaluar dados los objetivos del estudio es como el modelo AFC imputado se parece al modelo AFC original, y no si alcanza un buen ajuste o pesos factoriales adecuados, porque el modelo AFC original no los alcanza.

Por lo general podemos concluir que el método de imputación múltiple es eficaz porque recupera con bastante robusted el modelo AFC original introduciendo como input la matriz de varianzas y covarianzas. Los índices de ajuste, aunque en concreto TLI y CFI son los que más difieren (en el modelo AFC imputado interpretamos que no ajusta y en el modelo AFC original sí) no son valores muy dispares entre modelos y en términos muy generales, tienen un ajuste muy similar. Finalmente, respecto a los valores de los pesos factoriales y de las covarianzas entre factores, aunque puedan no coincidir, son diferencias mínimas y se llegan a las mismas interpretaciones.

Por último, aunque no hayamos podido identificar el problema que puede subyacer al modelo AFC original para obtener unos índices de ajuste bastante mejorables y unos pesos factoriales superiores a 1, nuestra sospecha es que se debe al tamaño muestral. El tamaño muestral del estudio es de 96 sujetos y la literatura recomienda emplear muestras de 300 sujetos (200 mínimo) para realizar Análisis Factoriales Confirmatorios robustos. Por ello, nuestra recomendación para futuros estudios es que se recoja un tamaño muestral adecuado, pero no excesivamente grande dado que chi-cuadrado es sensible al tamaño muestral.

ANEXO

"Script de R"

```
rm(list=ls())

set.seed(123)

library(haven)
```

```
## Warning: package 'haven' was built under R version 4.3.3
```

```
library(mice)
```

```
## Warning: package 'mice' was built under R version 4.3.3
```

```
##
## Attaching package: 'mice'
```

```
## The following object is masked from 'package:stats':
##
##     filter
```

```
## The following objects are masked from 'package:base':
##
##     cbind, rbind
```

```
library(reshape2)
library(mitml)
```

```
## Warning: package 'mitml' was built under R version 4.3.3

## *** This is beta software. Please report any bugs!
## *** See the NEWS file for recent changes.
```

```
library(lme4)
```

```
## Warning: package 'lme4' was built under R version 4.3.3

## Loading required package: Matrix

## Warning: package 'Matrix' was built under R version 4.3.3
```

```
library(lavaan)
```

```
## Warning: package 'lavaan' was built under R version 4.3.3

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

```
library(semPlot)
```

```
## Warning: package 'semPlot' was built under R version 4.3.2
```

```
library(psych)
```

```
## Warning: package 'psych' was built under R version 4.3.3

##
## Attaching package: 'psych'

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

```
datos <- read_sav("Liderazgo.sav")
View(datos)

table(is.na(datos))
```

```
##
## FALSE
##  1440
```

```
#obtenemos la matriz de var-cov de los datos  brutos para chequear luego como recupera la matriz la imp

datos <- datos[,-c(14,15,3,9)] #quitamos las variables de los factores y de los ítems que no forman par

raw_m <- cov(datos)
raw_m
```

```
##           p1       p2       p4       p5       p6       p7       p8      p10
## p1  2.410088 1.890132 1.274561 1.886184 1.463816 1.355921 1.430263 1.879605
## p2  1.890132 2.827961 1.225000 2.049671 1.546382 1.529276 1.628289 1.996382
## p4  1.274561 1.225000 2.756140 1.390789 1.340789 1.219737 1.002632 1.572368
## p5  1.886184 2.049671 1.390789 2.677961 1.962829 1.300987 1.669079 1.954934
## p6  1.463816 1.546382 1.340789 1.962829 3.591118 1.016118 1.591447 1.777961
## p7  1.355921 1.529276 1.219737 1.300987 1.016118 2.067434 1.257237 1.771382
## p8  1.430263 1.628289 1.002632 1.669079 1.591447 1.257237 2.027632 1.449342
## p10 1.879605 1.996382 1.572368 1.954934 1.777961 1.771382 1.449342 2.975329
## p11 1.992544 2.295395 1.520175 2.070395 1.984868 1.592763 1.640789 2.484868
## p12 1.674561 2.177632 1.082456 2.006579 1.767105 1.382895 1.771053 1.714474
## p13 1.799123 1.827632 1.580702 2.156579 2.448684 1.527632 1.744737 2.175000
##          p11      p12      p13
## p1  1.992544 1.674561 1.799123
## p2  2.295395 2.177632 1.827632
## p4  1.520175 1.082456 1.580702
## p5  2.070395 2.006579 2.156579
## p6  1.984868 1.767105 2.448684
## p7  1.592763 1.382895 1.527632
## p8  1.640789 1.771053 1.744737
## p10 2.484868 1.714474 2.175000
## p11 3.315351 2.257018 2.248246
## p12 2.257018 2.671930 1.938596
## p13 2.248246 1.938596 3.008772
```

```r
#Simulamos valores NA al 40% (MCAR)

datos_NA <- datos

table(is.na(datos_NA))
```

```
##
## FALSE
##  1056
```

```r
# Proporción de valores perdidos a introducir (puedes ajustar según tus necesidades)
proporcion_NA<- 0.4

# Obtener el número de filas y columnas en tus datos
num_filas <- nrow(datos_NA)
num_columnas <- ncol(datos_NA)

# Calcular el número total de valores a introducir como perdidos
num_NA <- round(proporcion_NA* num_filas * num_columnas)

# Generar ubicaciones aleatorias para los valores perdidos
ubicaciones_NA <- data.frame(
  fila = sample(1:num_filas, num_NA, replace = TRUE),
  columna = sample(1:num_columnas, num_NA, replace = TRUE)
)

# Introducir valores perdidos en las ubicaciones generadas
for (i in 1:num_NA) {
```

```
  datos_NA[ubicaciones_NA$fila[i], ubicaciones_NA$columna[i]] <- NA
}


table(is.na(datos_NA))
```

```
##
## FALSE  TRUE
##   711   345
```

```
write.csv(datos_NA,"liderazgo_NA.txt")

View(datos_NA)

str(datos_NA)
```

```
## tibble [96 x 11] (S3: tbl_df/tbl/data.frame)
##  $ p1 : num [1:96] 6 NA 6 2 6 5 NA 6 NA 7 ...
##   ..- attr(*, "label")= chr "Ayuda a cumplir los objetivos del trabajo"
##   ..- attr(*, "format.spss")= chr "F2.0"
##   ..- attr(*, "display_width")= int 4
##  $ p2 : num [1:96] 6 NA 6 2 5 4 5 6 NA 6 ...
##   ..- attr(*, "label")= chr "Da ejemplo cuando es necesario"
##   ..- attr(*, "format.spss")= chr "F2.0"
##   ..- attr(*, "display_width")= int 4
##  $ p4 : num [1:96] 6 7 6 2 6 5 NA NA 7 6 ...
##   ..- attr(*, "label")= chr "Realmente delega en sus colaboradores"
##   ..- attr(*, "format.spss")= chr "F2.0"
##   ..- attr(*, "display_width")= int 4
##  $ p5 : num [1:96] 6 6 6 NA 6 5 NA NA NA NA ...
##   ..- attr(*, "label")= chr "Ofrece nuevas soluciones a los problemas"
##   ..- attr(*, "format.spss")= chr "F2.0"
##   ..- attr(*, "display_width")= int 4
##  $ p6 : num [1:96] 7 6 6 NA 3 NA NA 3 1 6 ...
##   ..- attr(*, "label")= chr "Nos reúne con regularidad para fijar metas de trabajo"
##   ..- attr(*, "format.spss")= chr "F2.0"
##   ..- attr(*, "display_width")= int 4
##  $ p7 : num [1:96] 7 6 6 2 6 6 6 7 4 NA ...
##   ..- attr(*, "label")= chr "Establece objetivos alcanzables"
##   ..- attr(*, "format.spss")= chr "F2.0"
##   ..- attr(*, "display_width")= int 4
##  $ p8 : num [1:96] 7 6 NA 2 6 NA 7 NA 5 6 ...
##   ..- attr(*, "label")= chr "Busca la calidad en todas las tareas"
##   ..- attr(*, "format.spss")= chr "F2.0"
##   ..- attr(*, "display_width")= int 4
##  $ p10: num [1:96] NA NA 6 2 NA NA NA 7 NA NA ...
##   ..- attr(*, "label")= chr "Reconoce todos mis esfuerzos y logros"
##   ..- attr(*, "format.spss")= chr "F2.0"
##   ..- attr(*, "display_width")= int 5
##  $ p11: num [1:96] NA 6 6 1 5 2 NA NA NA 6 ...
##   ..- attr(*, "label")= chr "Busca que las recopensas sean justas"
##   ..- attr(*, "format.spss")= chr "F2.0"
##   ..- attr(*, "display_width")= int 5
```

```
##  $ p12: num [1:96] 7 NA 6 1 6 3 6 5 NA 6 ...
##   ..- attr(*, "label")= chr "Resuelve los conflictos con eficacia"
##   ..- attr(*, "format.spss")= chr "F2.0"
##   ..- attr(*, "display_width")= int 5
##  $ p13: num [1:96] 6 5 6 1 NA NA NA 6 2 6 ...
##   ..- attr(*, "label")= chr "Informa sobre los resultados del trabajo"
##   ..- attr(*, "format.spss")= chr "F2.0"
##   ..- attr(*, "display_width")= int 5
```

```r
#imputamos datos

imp <- mice(datos_NA, seed = 20000, meth = "pmm", m = 20)
```

```
##
##  iter imp variable
##   1   1  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   1   2  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   1   3  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   1   4  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   1   5  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   1   6  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   1   7  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   1   8  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   1   9  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   1   10  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   1   11  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   1   12  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   1   13  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   1   14  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   1   15  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   1   16  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   1   17  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   1   18  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   1   19  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   1   20  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   2   1  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   2   2  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   2   3  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   2   4  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   2   5  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   2   6  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   2   7  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   2   8  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   2   9  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   2   10  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   2   11  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   2   12  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   2   13  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   2   14  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   2   15  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   2   16  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   2   17  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   2   18  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   2   19  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
```

```
##   2   20  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   3   1  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   3   2  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   3   3  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   3   4  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   3   5  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   3   6  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   3   7  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   3   8  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   3   9  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   3   10  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   3   11  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   3   12  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   3   13  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   3   14  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   3   15  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   3   16  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   3   17  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   3   18  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   3   19  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   3   20  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   4   1  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   4   2  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   4   3  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   4   4  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   4   5  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   4   6  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   4   7  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   4   8  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   4   9  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   4   10  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   4   11  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   4   12  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   4   13  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   4   14  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   4   15  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   4   16  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   4   17  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   4   18  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   4   19  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   4   20  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   5   1  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   5   2  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   5   3  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   5   4  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   5   5  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   5   6  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   5   7  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   5   8  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   5   9  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   5   10  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   5   11  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   5   12  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   5   13  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
```

```
##   5   14  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   5   15  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   5   16  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   5   17  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   5   18  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   5   19  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
##   5   20  p1  p2  p4  p5  p6  p7  p8  p10  p11  p12  p13
```

```
imp$imp$p1  # Imputación del primer item en las 20 bases de datos
```

```
##    1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
## 2  6 7 6 6 6 6 6 6 6  7  6  6  6  6  7  6  7  6  7  6
## 7  5 7 5 6 7 6 6 6 4  6  7  4  5  6  6  6  5  6  6  7
## 9  5 5 5 6 5 6 5 4 5  4  5  1  3  6  6  2  4  3  4  2
## 14 6 6 6 7 7 6 6 6 7  6  6  7  6  7  7  6  6  6  7  7
## 24 5 6 7 6 5 6 6 6 5  5  7  7  3  6  6  5  7  7  6  7
## 26 5 2 2 2 1 3 1 1 2  3  1  2  1  1  2  2  3  2  2  1
## 27 6 6 7 6 6 6 7 5 6  6  7  7  7  7  6  6  6  6  7  7
## 29 6 6 5 6 5 7 6 6 7  5  6  6  5  4  4  6  6  6  6  6
## 30 6 7 6 4 7 7 7 6 6  6  7  6  6  6  6  6  6  6  6  6
## 32 6 7 7 4 7 6 7 5 7  6  6  6  6  7  6  6  6  6  7  7
## 36 6 3 3 4 4 3 2 4 3  5  5  2  5  5  4  4  5  5  3  4
## 42 4 5 5 3 5 5 5 4 6  4  2  5  5  5  4  5  4  4  4  5
## 45 6 5 6 5 6 5 5 6 4  6  4  3  6  6  5  5  5  6  5  6
## 47 3 5 5 5 5 5 4 3 4  2  5  4  4  4  4  5  6  5  6  4
## 48 6 7 6 7 6 6 6 6 6  7  7  7  6  6  6  7  6  6  6  6
## 53 3 5 5 4 4 5 2 5 5  5  5  5  5  5  5  5  3  5  5  4
## 54 6 6 6 5 7 6 7 6 6  6  6  6  7  6  7  6  6  7  6  6
## 60 5 5 4 5 5 5 2 4 3  4  5  3  4  4  5  5  3  5  4  4
## 74 5 6 5 5 5 5 5 5 5  5  5  6  7  5  4  4  6  5  3  4
## 76 2 5 4 1 1 2 3 2 3  4  3  2  2  1  4  2  4  2  1  1
## 79 6 6 5 5 5 5 6 6 4  4  6  6  6  5  4  7  6  4  6  6
## 81 6 6 7 6 7 7 6 7 6  6  6  7  6  6  6  6  6  6  6  6
## 86 5 5 4 5 5 3 5 5 5  5  2  1  4  4  6  5  2  1  4  1
## 90 6 5 5 6 5 5 6 6 6  6  5  6  7  4  5  4  6  5  7  3
## 93 6 5 6 6 6 7 6 6 7  6  7  6  6  7  6  6  6  6  7  7
## 94 5 6 6 4 6 4 6 5 6  7  5  5  4  7  6  5  5  2  5  5
## 95 2 5 3 1 3 4 5 1 4  5  5  3  1  6  5  5  5  5  2  3
## 96 4 4 4 4 5 2 5 5 5  5  5  5  5  3  5  4  3  1  6  3
```

```
complete_data <- complete(imp, 1)  # Imputación de la base de datos 1, los 13 items

matrices <- list()

for (i in 1:20) {
  data <- complete(imp, action = i)
  cov_matrix <- cov(data)
  matrices[[i]] <- cov_matrix
}

head(matrices)
```

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

```
##           p1       p2       p4       p5       p6       p7       p8      p10
## p1  2.229825 1.515789 1.505702 1.942105 1.725000 1.267105 1.524123 1.654825
## p2  1.515789 2.273684 1.263158 1.673684 1.631579 1.589474 1.210526 1.915789
## p4  1.505702 1.263158 2.668311 1.819737 1.713487 1.217434 1.600548 1.907346
## p5  1.942105 1.673684 1.819737 2.552632 1.877632 1.293421 1.764474 1.846053
## p6  1.725000 1.631579 1.713487 1.877632 4.146382 1.445066 1.911513 2.080592
## p7  1.267105 1.589474 1.217434 1.293421 1.445066 1.906908 1.244408 1.674013
## p8  1.524123 1.210526 1.600548 1.764474 1.911513 1.244408 2.149890 1.752741
## p10 1.654825 1.915789 1.907346 1.846053 2.080592 1.674013 1.752741 3.123575
## p11 1.634649 1.936842 1.517654 1.864474 2.781250 1.666776 1.501206 2.496162
## p12 1.869737 1.905263 1.572039 2.032895 2.460197 1.703618 2.018750 1.894408
## p13 1.711842 1.600000 1.550987 1.938158 2.733882 1.787829 1.976645 2.210197
##          p11      p12      p13
## p1  1.634649 1.869737 1.711842
## p2  1.936842 1.905263 1.600000
## p4  1.517654 1.572039 1.550987
## p5  1.864474 2.032895 1.938158
## p6  2.781250 2.460197 2.733882
## p7  1.666776 1.703618 1.787829
## p8  1.501206 2.018750 1.976645
## p10 2.496162 1.894408 2.210197
## p11 3.278838 2.259539 2.280592
## p12 2.259539 2.980592 2.212171
## p13 2.280592 2.212171 2.896382
##
## [[2]]
##           p1        p2        p4       p5       p6       p7       p8      p10
## p1  2.217434 1.6529605 1.3065789 1.593421 1.482237 1.167763 1.456579 1.896382
## p2  1.652961 2.5393640 0.8592105 1.721491 1.886623 1.363377 1.556579 1.756469
## p4  1.306579 0.8592105 2.8052632 1.478947 1.465789 1.334211 1.331579 1.443421
## p5  1.593421 1.7214912 1.4789474 2.419298 1.621930 1.114912 1.742105 1.895175
## p6  1.482237 1.8866228 1.4657895 1.621930 3.852193 1.437281 1.986842 1.973465
## p7  1.167763 1.3633772 1.3342105 1.114912 1.437281 1.799561 1.139474 1.634430
## p8  1.456579 1.5565789 1.3315789 1.742105 1.986842 1.139474 2.152632 1.572368
## p10 1.896382 1.7564693 1.4434211 1.895175 1.973465 1.634430 1.572368 2.957785
## p11 2.198026 2.2760965 1.2868421 2.190351 2.315351 1.437281 1.734211 2.562939
## p12 1.471711 2.0269737 0.9605263 1.744737 1.932895 1.209211 1.692105 1.619079
## p13 1.816447 1.7506579 1.5500000 1.702632 2.601316 1.530263 2.007895 2.121711
##          p11       p12      p13
## p1  2.198026 1.4717105 1.816447
## p2  2.276096 2.0269737 1.750658
## p4  1.286842 0.9605263 1.550000
## p5  2.190351 1.7447368 1.702632
## p6  2.315351 1.9328947 2.601316
## p7  1.437281 1.2092105 1.530263
## p8  1.734211 1.6921053 2.007895
## p10 2.562939 1.6190789 2.121711
## p11 3.494298 2.3223684 2.422368
## p12 2.322368 2.5118421 1.990789
## p13 2.422368 1.9907895 3.069737
##
## [[3]]
##           p1       p2        p4       p5       p6       p7       p8      p10
## p1  2.257785 1.608443 1.4560307 1.755811 1.553399 1.506360 1.443202 1.689035
```

```
## p2  1.608443 2.502522 1.2487939 1.881469 1.888268 1.747149 1.136623 2.102193
## p4  1.456031 1.248794 2.6893640 1.648794 1.670943 1.208114 1.587061 1.745175
## p5  1.755811 1.881469 1.6487939 2.586732 1.877741 1.431360 1.757675 1.439035
## p6  1.553399 1.888268 1.6709430 1.877741 4.020943 1.655482 1.908114 1.966228
## p7  1.506360 1.747149 1.2081140 1.431360 1.655482 2.115351 1.167982 1.762281
## p8  1.443202 1.136623 1.5870614 1.757675 1.908114 1.167982 2.346930 1.246491
## p10 1.689035 2.102193 1.7451754 1.439035 1.966228 1.762281 1.246491 3.008772
## p11 2.101535 2.413377 1.4734649 1.992325 2.573465 1.921491 1.574123 2.627193
## p12 1.721382 1.879934 0.9582237 1.753618 1.792434 1.617763 1.807237 1.480263
## p13 1.892325 1.722588 1.5695175 1.922588 2.374781 1.771491 1.739912 2.285088
##          p11       p12      p13
## p1  2.101535 1.7213816 1.892325
## p2  2.413377 1.8799342 1.722588
## p4  1.473465 0.9582237 1.569518
## p5  1.992325 1.7536184 1.922588
## p6  2.573465 1.7924342 2.374781
## p7  1.921491 1.6177632 1.771491
## p8  1.574123 1.8072368 1.739912
## p10 2.627193 1.4802632 2.285088
## p11 3.452193 2.3480263 2.465351
## p12 2.348026 2.6569079 1.796711
## p13 2.465351 1.7967105 2.852193
##
## [[4]]
##           p1       p2        p4       p5       p6       p7       p8      p10
## p1  2.503509 1.839474 1.5824561 2.020175 1.904386 1.346930 1.484649 1.508772
## p2  1.839474 2.480263 1.2421053 1.914474 1.630263 1.636184 1.363816 1.600000
## p4  1.582456 1.242105 2.4771930 1.491228 1.564912 1.350877 1.017544 1.687719
## p5  2.020175 1.914474 1.4912281 2.504825 1.883772 1.457675 1.700219 1.596491
## p6  1.904386 1.630263 1.5649123 1.883772 3.873246 1.483991 1.579167 1.943860
## p7  1.346930 1.636184 1.3508772 1.457675 1.483991 2.078838 1.315899 1.777193
## p8  1.484649 1.363816 1.0175439 1.700219 1.579167 1.315899 1.899890 1.180702
## p10 1.508772 1.600000 1.6877193 1.596491 1.943860 1.777193 1.180702 3.045614
## p11 1.985526 2.130921 1.3894737 1.955921 2.687500 1.575329 1.387829 2.300000
## p12 1.877193 2.052632 0.9824561 1.912281 2.059649 1.587719 1.738596 1.361404
## p13 1.839912 1.665132 1.4175439 2.098904 2.651535 1.894189 1.774232 2.233333
##          p11       p12      p13
## p1  1.985526 1.8771930 1.839912
## p2  2.130921 2.0526316 1.665132
## p4  1.389474 0.9824561 1.417544
## p5  1.955921 1.9122807 2.098904
## p6  2.687500 2.0596491 2.651535
## p7  1.575329 1.5877193 1.894189
## p8  1.387829 1.7385965 1.774232
## p10 2.300000 1.3614035 2.233333
## p11 3.254276 2.3842105 2.229276
## p12 2.384211 2.7929825 1.980702
## p13 2.229276 1.9807018 2.970943
##
## [[5]]
##           p1        p2        p4       p5       p6        p7       p8      p10
## p1  2.498246 1.7842105 1.1868421 2.065789 1.850000 1.4552632 1.471053 2.002632
## p2  1.784211 2.3118421 0.8677632 1.926974 1.667763 1.7401316 1.312500 1.926974
## p4  1.186842 0.8677632 2.2016447 1.211513 1.285855 0.9207237 1.379276 1.443092
```

```
## p5  2.065789 1.9269737 1.2115132 2.659539 1.927303 1.5398026 1.644408 1.691118
## p6  1.850000 1.6677632 1.2858553 1.927303 4.222697 1.6786184 1.737171 2.200987
## p7  1.455263 1.7401316 0.9207237 1.539803 1.678618 2.2148026 1.200987 1.792434
## p8  1.471053 1.3125000 1.3792763 1.644408 1.737171 1.2009868 2.141118 1.654934
## p10 2.002632 1.9269737 1.4430921 1.691118 2.200987 1.7924342 1.654934 2.954276
## p11 2.289474 2.2697368 0.9151316 2.037500 2.620395 1.9243421 1.738816 2.521711
## p12 1.833333 1.8223684 0.8282895 1.924342 1.859868 1.7164474 1.630921 1.576974
## p13 2.116667 1.8282895 1.3266447 2.212829 2.674013 1.9588816 1.951645 2.402303
##          p11       p12      p13
## p1  2.2894737 1.8333333 2.116667
## p2  2.2697368 1.8223684 1.828289
## p4  0.9151316 0.8282895 1.326645
## p5  2.0375000 1.9243421 2.212829
## p6  2.6203947 1.8598684 2.674013
## p7  1.9243421 1.7164474 1.958882
## p8  1.7388158 1.6309211 1.951645
## p10 2.5217105 1.5769737 2.402303
## p11 3.3855263 2.4328947 2.517763
## p12 2.4328947 2.5469298 1.911623
## p13 2.5177632 1.9116228 2.976206
##
## [[6]]
##          p1       p2       p4       p5       p6       p7       p8       p10
## p1  2.386732 1.613816 1.204057 1.879167 1.541009 1.427303 1.529167 1.812171
## p2  1.613816 2.206579 1.073026 1.698684 1.064474 1.638816 1.040789 1.861184
## p4  1.204057 1.073026 2.368311 1.486623 1.314254 1.002961 1.173465 1.368092
## p5  1.879167 1.698684 1.486623 2.578509 1.633772 1.450658 1.836404 1.691447
## p6  1.541009 1.064474 1.314254 1.633772 3.536404 1.567763 1.681140 1.469079
## p7  1.427303 1.638816 1.002961 1.450658 1.567763 2.014803 1.211184 1.861513
## p8  1.529167 1.040789 1.173465 1.836404 1.681140 1.211184 2.294298 1.330921
## p10 1.812171 1.861184 1.368092 1.691447 1.469079 1.861513 1.330921 2.725329
## p11 2.477851 2.053947 1.261623 2.128509 1.820614 1.704605 1.807456 2.342763
## p12 1.895724 1.744079 1.029276 1.955921 1.736184 1.577961 1.821711 1.645724
## p13 1.767654 1.459868 1.431031 2.064693 2.476535 2.035855 2.046272 2.177303
##          p11      p12      p13
## p1  2.477851 1.895724 1.767654
## p2  2.053947 1.744079 1.459868
## p4  1.261623 1.029276 1.431031
## p5  2.128509 1.955921 2.064693
## p6  1.820614 1.736184 2.476535
## p7  1.704605 1.577961 2.035855
## p8  1.807456 1.821711 2.046272
## p10 2.342763 1.645724 2.177303
## p11 3.515351 2.494079 2.097588
## p12 2.494079 2.699013 1.904276
## p13 2.097588 1.904276 3.291996
```

```r
# Ahora promedio las matrices de covarianza
average_cov_matrix <- Reduce(`+`, matrices) / length(matrices)


#################Recuperación de la matriz (matriz residual)###################

#raw_m[upper.tri(raw_m)] <- 0
```

```r
#raw_m

#average_cov_matrix[upper.tri(average_cov_matrix)] <- 0
#average_cov_matrix

#m_dif <- raw_m-average_cov_matrix  #(residuos muy bajos,la imputacion recupera bien)
#m_dif
##############################################################################



# Definiendo el modelo de CFA

modelo <- '

  factor1 =~ p1 + p2 + p8 + p5 + p12
  factor2 =~ p4 + p7 + p10 + p11
  factor3 =~ p6 + p13

  factor1 ~~ factor2
  factor1 ~~ factor3
  factor2 ~~ factor3
'

# Realizando el CFA con la matriz de covarianza promediada

afc <- cfa(modelo, sample.cov = average_cov_matrix, sample.nobs = 96, std.lv = TRUE, meanstructure = TRU
```

```
## Warning in lavaan::lavaan(model = modelo, sample.cov = average_cov_matrix, : lavaan WARNING:
##     sample.mean= argument is missing, but model contains
##     mean/intercept parameters.
```

```r
summary(afc, standardized=TRUE, fit.measures=TRUE)
```

```
## lavaan 0.6.17 ended normally after 29 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                        36
##
##   Number of observations                            96
##
## Model Test User Model:
##
##   Test statistic                               172.399
##   Degrees of freedom                                41
##   P-value (Chi-square)                           0.000
##
## Model Test Baseline Model:
##
##   Test statistic                              1047.698
##   Degrees of freedom                                55
```

```
##   P-value                                        0.000
##
## User Model versus Baseline Model:
##
##   Comparative Fit Index (CFI)                    0.868
##   Tucker-Lewis Index (TLI)                       0.822
##
## Loglikelihood and Information Criteria:
##
##   Loglikelihood user model (H0)              -1582.997
##   Loglikelihood unrestricted model (H1)      -1496.797
##
##   Akaike (AIC)                                3237.993
##   Bayesian (BIC)                              3330.310
##   Sample-size adjusted Bayesian (SABIC)       3216.642
##
## Root Mean Square Error of Approximation:
##
##   RMSEA                                          0.183
##   90 Percent confidence interval - lower         0.155
##   90 Percent confidence interval - upper         0.211
##   P-value H_0: RMSEA <= 0.050                    0.000
##   P-value H_0: RMSEA >= 0.080                    1.000
##
## Standardized Root Mean Square Residual:
##
##   SRMR                                           0.047
##
## 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
##   factor1 =~
##     p1                1.326    0.129   10.255    0.000    1.326    0.850
##     p2                1.322    0.129   10.228    0.000    1.322    0.849
##     p8                1.166    0.128    9.104    0.000    1.166    0.786
##     p5                1.393    0.131   10.646    0.000    1.393    0.870
##     p12               1.392    0.134   10.358    0.000    1.392    0.855
##   factor2 =~
##     p4                1.048    0.152    6.892    0.000    1.048    0.643
##     p7                1.140    0.119    9.579    0.000    1.140    0.816
##     p10               1.469    0.144   10.180    0.000    1.469    0.848
##     p11               1.607    0.151   10.670    0.000    1.607    0.874
##   factor3 =~
##     p6                1.557    0.172    9.028    0.000    1.557    0.789
##     p13               1.639    0.136   12.063    0.000    1.639    0.958
##
## Covariances:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##   factor1 ~~
```

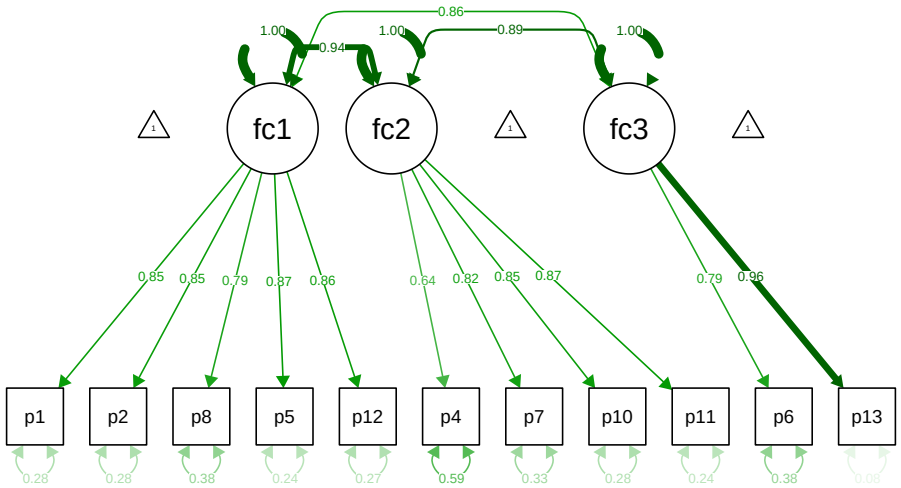
```
##     factor2           0.944    0.024   39.471    0.000    0.944    0.944
##     factor3           0.865    0.039   22.183    0.000    0.865    0.865
##   factor2 ~~
##     factor3           0.891    0.038   23.667    0.000    0.891    0.891
##
## Intercepts:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##    .p1                0.000    0.159    0.000    1.000    0.000    0.000
##    .p2                0.000    0.159    0.000    1.000    0.000    0.000
##    .p8                0.000    0.151    0.000    1.000    0.000    0.000
##    .p5                0.000    0.163    0.000    1.000    0.000    0.000
##    .p12               0.000    0.166    0.000    1.000    0.000    0.000
##    .p4                0.000    0.166    0.000    1.000    0.000    0.000
##    .p7                0.000    0.143    0.000    1.000    0.000    0.000
##    .p10               0.000    0.177    0.000    1.000    0.000    0.000
##    .p11               0.000    0.188    0.000    1.000    0.000    0.000
##    .p6                0.000    0.202    0.000    1.000    0.000    0.000
##    .p13               0.000    0.175    0.000    1.000    0.000    0.000
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##    .p1                0.676    0.115    5.882    0.000    0.676    0.278
##    .p2                0.680    0.115    5.894    0.000    0.680    0.280
##    .p8                0.839    0.134    6.281    0.000    0.839    0.382
##    .p5                0.624    0.110    5.673    0.000    0.624    0.243
##    .p12               0.712    0.122    5.832    0.000    0.712    0.268
##    .p4                1.561    0.236    6.605    0.000    1.561    0.587
##    .p7                0.653    0.109    5.986    0.000    0.653    0.335
##    .p10               0.842    0.148    5.692    0.000    0.842    0.280
##    .p11               0.801    0.150    5.344    0.000    0.801    0.237
##    .p6                1.473    0.250    5.896    0.000    1.473    0.378
##    .p13               0.240    0.149    1.608    0.108    0.240    0.082
##     factor1           1.000                               1.000    1.000
##     factor2           1.000                               1.000    1.000
##     factor3           1.000                               1.000    1.000
```

```
coef(afc)
```

```
##      factor1=~p1      factor1=~p2      factor1=~p8      factor1=~p5
##            1.326            1.322            1.166            1.393
##     factor1=~p12      factor2=~p4      factor2=~p7     factor2=~p10
##            1.392            1.048            1.140            1.469
##     factor2=~p11      factor3=~p6     factor3=~p13 factor1~~factor2
##            1.607            1.557            1.639            0.944
## factor1~~factor3 factor2~~factor3          p1~~p1           p2~~p2
##            0.865            0.891            0.676            0.680
##           p8~~p8           p5~~p5         p12~~p12           p4~~p4
##            0.839            0.624            0.712            1.561
##           p7~~p7         p10~~p10         p11~~p11           p6~~p6
##            0.653            0.842            0.801            1.473
##         p13~~p13             p1~1             p2~1             p8~1
##            0.240            0.000            0.000            0.000
##             p5~1            p12~1             p4~1             p7~1
##            0.000            0.000            0.000            0.000
```

```
##           p10~1           p11~1            p6~1           p13~1
##           0.000           0.000           0.000           0.000
```

```
semPaths(afc,"std",edge.label.cex=0.5, curvePivot = TRUE)
```




```
############### Realizamos también el CFA con los datos brutos (sin simular datos MCAR) e introduciendo
```

```
raw_datos <- read_sav("Liderazgo.sav")
str(raw_datos)
```

```
## tibble [96 x 15] (S3: tbl_df/tbl/data.frame)
##  $ p1      : num [1:96] 6 6 6 2 6 5 7 6 6 7 ...
##   ..- attr(*, "label")= chr "Ayuda a cumplir los objetivos del trabajo"
##   ..- attr(*, "format.spss")= chr "F2.0"
##   ..- attr(*, "display_width")= int 4
##  $ p2      : num [1:96] 6 6 6 2 5 4 5 6 4 6 ...
##   ..- attr(*, "label")= chr "Da ejemplo cuando es necesario"
##   ..- attr(*, "format.spss")= chr "F2.0"
##   ..- attr(*, "display_width")= int 4
##  $ p3      : num [1:96] 5 6 6 2 3 5 6 6 4 6 ...
##   ..- attr(*, "label")= chr "Explica las razones de sus decisiones"
##   ..- attr(*, "format.spss")= chr "F2.0"
##   ..- attr(*, "display_width")= int 4
##  $ p4      : num [1:96] 6 7 6 2 6 5 5 6 7 6 ...
```

```
##   ..- attr(*, "label")= chr "Realmente delega en sus colaboradores"
##   ..- attr(*, "format.spss")= chr "F2.0"
##   ..- attr(*, "display_width")= int 4
##  $ p5      : num [1:96] 6 6 6 2 6 5 5 5 2 6 ...
##   ..- attr(*, "label")= chr "Ofrece nuevas soluciones a los problemas"
##   ..- attr(*, "format.spss")= chr "F2.0"
##   ..- attr(*, "display_width")= int 4
##  $ p6      : num [1:96] 7 6 6 3 3 3 7 3 1 6 ...
##   ..- attr(*, "label")= chr "Nos reúne con regularidad para fijar metas de trabajo"
##   ..- attr(*, "format.spss")= chr "F2.0"
##   ..- attr(*, "display_width")= int 4
##  $ p7      : num [1:96] 7 6 6 2 6 6 6 7 4 6 ...
##   ..- attr(*, "label")= chr "Establece objetivos alcanzables"
##   ..- attr(*, "format.spss")= chr "F2.0"
##   ..- attr(*, "display_width")= int 4
##  $ p8      : num [1:96] 7 6 6 2 6 4 7 7 5 6 ...
##   ..- attr(*, "label")= chr "Busca la calidad en todas las tareas"
##   ..- attr(*, "format.spss")= chr "F2.0"
##   ..- attr(*, "display_width")= int 4
##  $ p9      : num [1:96] 6 6 6 3 6 6 7 7 7 6 ...
##   ..- attr(*, "label")= chr "Es respetado profesionalmente"
##   ..- attr(*, "format.spss")= chr "F2.0"
##   ..- attr(*, "display_width")= int 4
##  $ p10     : num [1:96] 6 6 6 2 5 3 5 7 2 6 ...
##   ..- attr(*, "label")= chr "Reconoce todos mis esfuerzos y logros"
##   ..- attr(*, "format.spss")= chr "F2.0"
##   ..- attr(*, "display_width")= int 5
##  $ p11     : num [1:96] 6 6 6 1 5 2 4 5 3 6 ...
##   ..- attr(*, "label")= chr "Busca que las recopensas sean justas"
##   ..- attr(*, "format.spss")= chr "F2.0"
##   ..- attr(*, "display_width")= int 5
##  $ p12     : num [1:96] 7 6 6 1 6 3 6 5 2 6 ...
##   ..- attr(*, "label")= chr "Resuelve los conflictos con eficacia"
##   ..- attr(*, "format.spss")= chr "F2.0"
##   ..- attr(*, "display_width")= int 5
##  $ p13     : num [1:96] 6 5 6 1 4 5 6 6 2 6 ...
##   ..- attr(*, "label")= chr "Informa sobre los resultados del trabajo"
##   ..- attr(*, "format.spss")= chr "F2.0"
##   ..- attr(*, "display_width")= int 5
##  $ satisfac: dbl+lbl [1:96]  9,  9,  9,  2,  8,  6,  9,  9,  5,  9,  9,  4, 10,  9,...
##    ..@ label        : chr "Satisfacción"
##    ..@ format.spss  : chr "F2.0"
##    ..@ display_width: int 10
##    ..@ labels       : Named num [1:10] 1 2 3 4 5 6 7 8 9 10
##    .. ..- attr(*, "names")= chr [1:10] "1" "2" "3" "4" ...
##  $ corte   : dbl+lbl [1:96] 2, 2, 2, 1, 2, 1, 2, 2, 1, 2, 2, 1, 2, 2, 2, 2, 2, 2, 1...
##    ..@ label        : chr "Corte satisfacción"
##    ..@ format.spss  : chr "F8.0"
##    ..@ display_width: int 10
##    ..@ labels       : Named num [1:2] 1 2
##    .. ..- attr(*, "names")= chr [1:2] "Bajo" "Alto"
```

```
#eliminamos variables innecesarias y las que no estan incluidas en el modelo
```

```r
raw_datos <- raw_datos[,-c(14,15,3,9)]

head(raw_datos)
```

```
## # A tibble: 6 x 11
##      p1    p2    p4    p5    p6    p7    p8   p10   p11   p12   p13
##   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1     6     6     6     6     7     7     7     6     6     7     6
## 2     6     6     7     6     6     6     6     6     6     6     5
## 3     6     6     6     6     6     6     6     6     6     6     6
## 4     2     2     2     2     3     2     2     2     1     1     1
## 5     6     5     6     6     3     6     6     5     5     6     4
## 6     5     4     5     5     3     6     4     3     2     3     5
```

```r
matriz <- cov(raw_datos)

matriz
```

```
##           p1       p2       p4       p5       p6       p7       p8      p10
## p1  2.410088 1.890132 1.274561 1.886184 1.463816 1.355921 1.430263 1.879605
## p2  1.890132 2.827961 1.225000 2.049671 1.546382 1.529276 1.628289 1.996382
## p4  1.274561 1.225000 2.756140 1.390789 1.340789 1.219737 1.002632 1.572368
## p5  1.886184 2.049671 1.390789 2.677961 1.962829 1.300987 1.669079 1.954934
## p6  1.463816 1.546382 1.340789 1.962829 3.591118 1.016118 1.591447 1.777961
## p7  1.355921 1.529276 1.219737 1.300987 1.016118 2.067434 1.257237 1.771382
## p8  1.430263 1.628289 1.002632 1.669079 1.591447 1.257237 2.027632 1.449342
## p10 1.879605 1.996382 1.572368 1.954934 1.777961 1.771382 1.449342 2.975329
## p11 1.992544 2.295395 1.520175 2.070395 1.984868 1.592763 1.640789 2.484868
## p12 1.674561 2.177632 1.082456 2.006579 1.767105 1.382895 1.771053 1.714474
## p13 1.799123 1.827632 1.580702 2.156579 2.448684 1.527632 1.744737 2.175000
##          p11      p12      p13
## p1  1.992544 1.674561 1.799123
## p2  2.295395 2.177632 1.827632
## p4  1.520175 1.082456 1.580702
## p5  2.070395 2.006579 2.156579
## p6  1.984868 1.767105 2.448684
## p7  1.592763 1.382895 1.527632
## p8  1.640789 1.771053 1.744737
## p10 2.484868 1.714474 2.175000
## p11 3.315351 2.257018 2.248246
## p12 2.257018 2.671930 1.938596
## p13 2.248246 1.938596 3.008772
```

```r
#CFA

raw_modelo <- '

  factor1=~p1+p2+p8+p5+p12
  factor2=~p4+p7+p10+p11
  factor3=~p6+p13

  factor1~~factor2
```

```
  factor1~~factor3
  factor2~~factor3

'


raw_afc <- cfa(raw_modelo, sample.cov = matriz, sample.nobs = 96, std.lv = TRUE, meanstructure = TRUE, 
```

```
## Warning in lavaan::lavaan(model = raw_modelo, sample.cov = matriz, sample.nobs = 96, : lavaan WARNIN
##     sample.mean= argument is missing, but model contains
##     mean/intercept parameters.
```

```
summary(raw_afc, standardized=TRUE, fit.measures=TRUE)
```

```
## lavaan 0.6.17 ended normally after 28 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                        36
##
##   Number of observations                            96
##
## Model Test User Model:
##
##   Test statistic                                80.412
##   Degrees of freedom                                41
##   P-value (Chi-square)                           0.000
##
## Model Test Baseline Model:
##
##   Test statistic                               938.310
##   Degrees of freedom                                55
##   P-value                                        0.000
##
## User Model versus Baseline Model:
##
##   Comparative Fit Index (CFI)                    0.955
##   Tucker-Lewis Index (TLI)                       0.940
##
## Loglikelihood and Information Criteria:
##
##   Loglikelihood user model (H0)              -1592.065
##   Loglikelihood unrestricted model (H1)      -1551.859
##
##   Akaike (AIC)                                3256.131
##   Bayesian (BIC)                              3348.447
##   Sample-size adjusted Bayesian (SABIC)       3234.780
##
## Root Mean Square Error of Approximation:
##
##   RMSEA                                          0.100
##   90 Percent confidence interval - lower         0.067
```

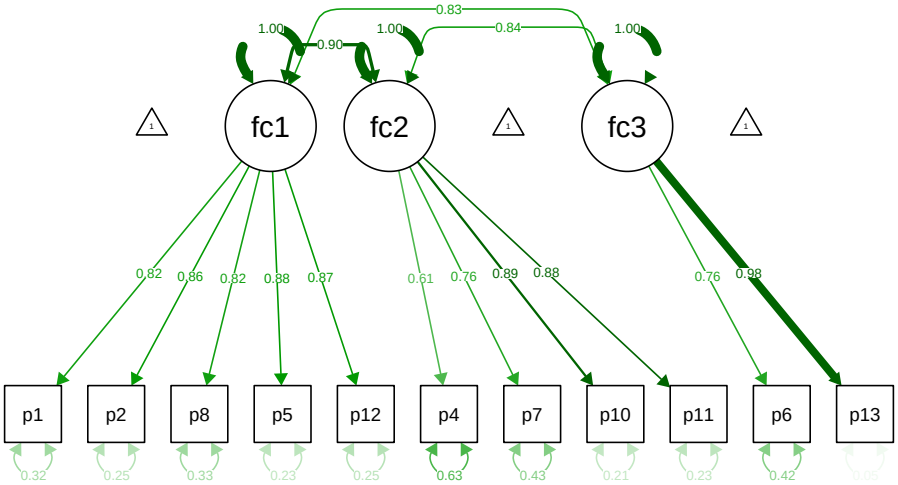
```
##   90 Percent confidence interval - upper         0.132
##   P-value H_0: RMSEA <= 0.050                    0.009
##   P-value H_0: RMSEA >= 0.080                    0.853
##
## Standardized Root Mean Square Residual:
##
##   SRMR                                           0.037
##
## Parameter Estimates:
##
##   Standard errors                             Standard
##   Information                                 Expected
##   Information saturated (h1) model          Structured
##
## Latent Variables:
##                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##   factor1 =~
##     p1               1.271    0.130    9.744    0.000    1.271    0.823
##     p2               1.445    0.137   10.527    0.000    1.445    0.864
##     p8               1.160    0.120    9.667    0.000    1.160    0.819
##     p5               1.430    0.132   10.819    0.000    1.430    0.879
##     p12              1.410    0.133   10.594    0.000    1.410    0.867
##   factor2 =~
##     p4               1.009    0.157    6.439    0.000    1.009    0.611
##     p7               1.084    0.127    8.566    0.000    1.084    0.758
##     p10              1.527    0.139   10.961    0.000    1.527    0.890
##     p11              1.595    0.148   10.770    0.000    1.595    0.880
##   factor3 =~
##     p6               1.439    0.168    8.561    0.000    1.439    0.763
##     p13              1.684    0.138   12.226    0.000    1.684    0.976
##
## Covariances:
##                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##   factor1 ~~
##     factor2          0.905    0.030   30.471    0.000    0.905    0.905
##     factor3          0.830    0.044   18.665    0.000    0.830    0.830
##   factor2 ~~
##     factor3          0.836    0.046   18.318    0.000    0.836    0.836
##
## Intercepts:
##                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##    .p1               0.000    0.158    0.000    1.000    0.000    0.000
##    .p2               0.000    0.171    0.000    1.000    0.000    0.000
##    .p8               0.000    0.145    0.000    1.000    0.000    0.000
##    .p5               0.000    0.166    0.000    1.000    0.000    0.000
##    .p12              0.000    0.166    0.000    1.000    0.000    0.000
##    .p4               0.000    0.169    0.000    1.000    0.000    0.000
##    .p7               0.000    0.146    0.000    1.000    0.000    0.000
##    .p10              0.000    0.175    0.000    1.000    0.000    0.000
##    .p11              0.000    0.185    0.000    1.000    0.000    0.000
##    .p6               0.000    0.192    0.000    1.000    0.000    0.000
##    .p13              0.000    0.176    0.000    1.000    0.000    0.000
##
## Variances:
```

```
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##    .p1                0.770    0.127    6.076    0.000    0.770    0.323
##    .p2                0.709    0.124    5.724    0.000    0.709    0.253
##    .p8                0.662    0.108    6.103    0.000    0.662    0.330
##    .p5                0.605    0.109    5.543    0.000    0.605    0.228
##    .p12               0.655    0.115    5.685    0.000    0.655    0.248
##    .p4                1.710    0.258    6.618    0.000    1.710    0.627
##    .p7                0.871    0.140    6.217    0.000    0.871    0.426
##    .p10               0.613    0.126    4.846    0.000    0.613    0.208
##    .p11               0.738    0.146    5.049    0.000    0.738    0.225
##    .p6                1.484    0.250    5.939    0.000    1.484    0.418
##    .p13               0.140    0.178    0.789    0.430    0.140    0.047
##     factor1           1.000                              1.000    1.000
##     factor2           1.000                              1.000    1.000
##     factor3           1.000                              1.000    1.000
```

```
coef(raw_afc)
```

```
##      factor1=~p1      factor1=~p2      factor1=~p8      factor1=~p5
##            1.271            1.445            1.160            1.430
##     factor1=~p12      factor2=~p4      factor2=~p7     factor2=~p10
##            1.410            1.009            1.084            1.527
##     factor2=~p11      factor3=~p6     factor3=~p13 factor1~~factor2
##            1.595            1.439            1.684            0.905
## factor1~~factor3 factor2~~factor3          p1~~p1           p2~~p2
##            0.830            0.836            0.770            0.709
##           p8~~p8           p5~~p5         p12~~p12           p4~~p4
##            0.662            0.605            0.655            1.710
##           p7~~p7         p10~~p10         p11~~p11           p6~~p6
##            0.871            0.613            0.738            1.484
##         p13~~p13             p1~1             p2~1             p8~1
##            0.140            0.000            0.000            0.000
##             p5~1            p12~1             p4~1             p7~1
##            0.000            0.000            0.000            0.000
##            p10~1            p11~1             p6~1            p13~1
##            0.000            0.000            0.000            0.000
```

```
semPaths(raw_afc,"std",edge.label.cex=0.5, curvePivot = TRUE)
```


0.83
1.00
0.90
1.00
0.84
1.00
fc1
fc2
fc3
0.82
0.86
0.82
0.88
0.87
0.61
0.76
0.89
0.88
0.76
0.98
p1
p2
p8
p5
p12
p4
p7
p10
p11
p6
p13
0.32
0.25
0.33
0.23
0.25
0.63
0.43
0.21
0.23
0.42