Guía para ACP y AF

Miguel Hernández

11/8/2019

Introducción

Antes de realizar un análisis, lo importante es tener clara la formulación del problema. En este caso trabajaremos con investigar con detectar las materias de matemáticas y ciencias naturales pertenecen a un grupo distinto que materias de francés y latín.

En el cuadro 1 se muestran las notas de las materias de 8 alumnos de la siguiente manera:

Cuadro 1: Calificaciones de materias

Matemáticas	Naturales	Francés	Latín
9	8	6	7
10	9	10	10
3	5	9	8
9	9	8	8
7	6	3	5
5	5	5	5
5	5	7	6
4	4	3	4

Proceso en R

Antes de realizar cualquier operación, hay que asegurarse que se tengan instalados los siguientes programas:

```
# Nombre de los paquetes que se utilizarán
paquetes = c("psych", "ggplot2", "ggcorrplot")

# Se crea un objeto con los nombres de los paquetes que no están instalados
nuevos_paquetes = paquetes[!paquetes %in% installed.packages()[,1]]

# Instala aquellos que no están instalados.
# Si no hay ninguno por instalar, no hace nada.
if(length(nuevos_paquetes)) install.packages(nuevos_paquetes)
```

Exploración de datos

Importamos datos que vienen en formato SPSS y los depositamos en el objeto `CL`.

```
CL = foreign::read.spss("data/Ciencias-Letras TOY EJEMPLO.sav", to.data.frame = T)
names(CL) = c("Matemáticas", names(CL)[2:4])
str(CL)
```

```
## 'data.frame':    8 obs. of  4 variables:
##  $ Matemáticas: num  9 10 3 9 7 5 5 4
##  $ Naturales  : num  8 9 5 9 6 5 5 4
##  $ Francés    : num  6 10 9 8 3 5 7 3
##  $ Latín      : num  7 10 8 8 5 5 6 4
```

```
##  - attr(*, "variable.labels")= Named chr  "Matematicas" "C Naturales" "Francés" "Latín"
##   ..- attr(*, "names")= chr  "Matematicas" "Naturales" "Francés" "Latín"
##  - attr(*, "codepage")= int 65001
```

El objeto `CL` es una `data.frame` con 8 observaciones y 4 variables, sin ningún valor faltante. Luego, calculamos la media y la desviación estándar de cada variable.

```
# Media
summary(CL)
```

```
##   Matemáticas      Naturales        Francés          Latín
##  Min.   : 3.00   Min.   :4.000   Min.   : 3.000   Min.   : 4.000
##  1st Qu.: 4.75   1st Qu.:5.000   1st Qu.: 4.500   1st Qu.: 5.000
##  Median : 6.00   Median :5.500   Median : 6.500   Median : 6.500
##  Mean   : 6.50   Mean   :6.375   Mean   : 6.375   Mean   : 6.625
##  3rd Qu.: 9.00   3rd Qu.:8.250   3rd Qu.: 8.250   3rd Qu.: 8.000
##  Max.   :10.00   Max.   :9.000   Max.   :10.000   Max.   :10.000
```

```
# Desviación estándar
apply(CL, 2, sd)
```

```
## Matemáticas   Naturales     Francés       Latín
##    2.618615    1.995531    2.615203    1.995531
```

Relación entre variables

La primera aproximación es ver si existen relaciones entre las variables que esperamos que pertenezcan a sus grupos. En este caso, Matemáticas y Naturales están correlacionados; análogamente Francés y Latín. El paquete `Hmisc` ofrece la función `rcorr()` para ver una matriz de correlaciones con sus significancias estadísticas. Para ello, se necesita convertir `CL` a datos de clase `matrix`.

```
CL_cor = Hmisc::rcorr(as.matrix(CL))
cor_mat = CL_cor$r # Matriz de correlaciones
sig_mat = CL_cor$P # Matriz de p-valores de la correlación
```

También se puede ver gráficamente mediante el paquete `ggcorrplot`, en donde se pone la matriz de correlaciones `cor_mat`, como se muestra en la gráfica 1.

```
library(ggcorrplot)
p.mat = cor_pmat(CL) # Otra forma de calcular la matriz de p-valores
ggcorrplot(
  cor_mat,
  hc.order = TRUE,
  type = "lower",
  lab = TRUE,
  colors = c("#ee5200", "white", "#009cee"),
  p.mat = p.mat, # Habilitar ver las correlaciones no significativas
  pch = 0, # Los p-valores no significativos se muestran como cuadros
  pch.cex = 12
)
```

Otra forma de verlo es ver si en general las variables están relacionadas mediante el determinante aplicando la función `det()` a la matriz de correlaciones ubicada en `cor_mat`. Si el determinante se acerca a 0, significa que hay relación entre las variables y podemos seguir explorando si el análisis de factores es adecuado.

```
det(cor_mat)
```

```
## [1] 0.001294676
```

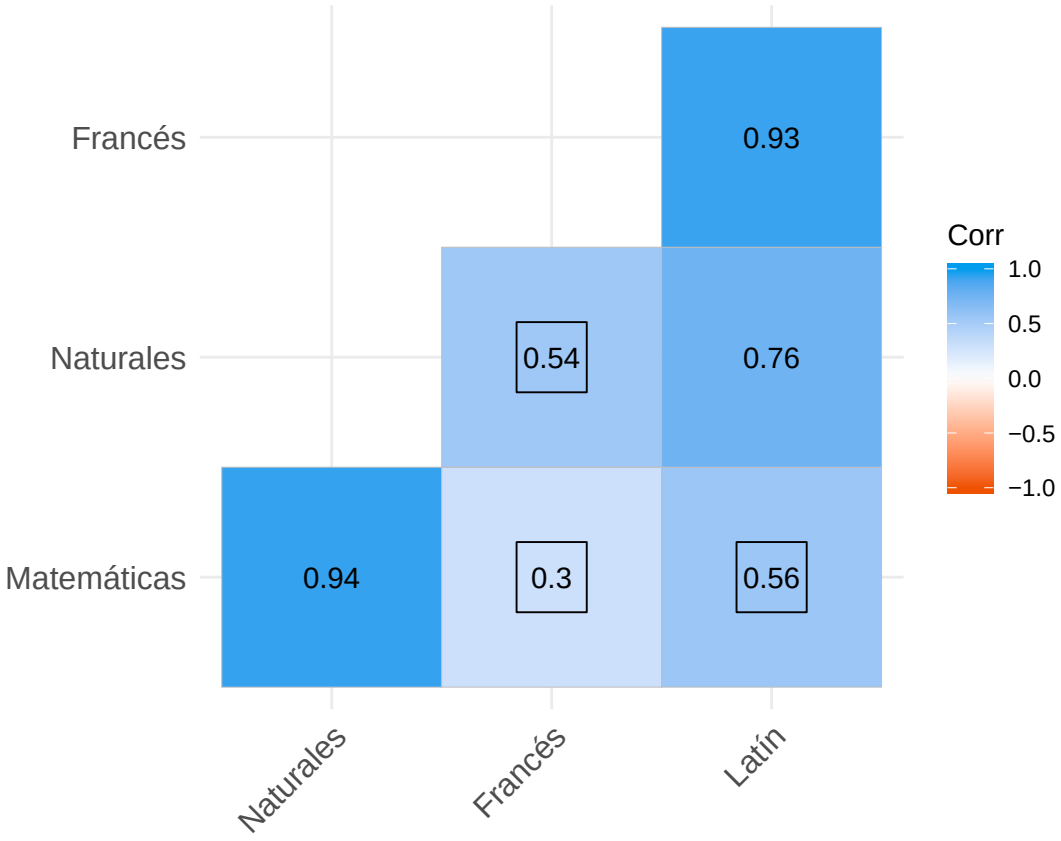

Figura 1: Matriz de correlaciones de materias

El determinante se acerca a cero y podemos continuar.

Prueba de esfericidad

Queremos descartar si los datos tienen una forma esférica. Es decir, una esfericidad complete se presenta mediante la matriz:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix}$$

Entonces, para descartar que los datos se comportan de esa forma, utilizamos la prueba de esfericidad de Bartlett, en donde la H_0 es que los datos presentan esfericidad completa. Es decir, mide el grado en que la matriz se desvía de la matriz de identidad $\mathbf{R}$. Para ello usamos la función `cortest.bartlett()` del paquete `psych`. Los argumentos que requiere esta función son la matriz de correlaciones `cor_mat` y el número de observaciones, que lo podemos calcular con `nrow()`.

```
psych::cortest.bartlett(cor_mat, nrow(CL))
```

```
## $chisq
## [1] 32.13922
##
## $p.value
## [1] 1.53448e-05
##
## $df
## [1] 6
```

Dado que el p-valor es < 0.05, podemos decir que tenemos evidencia suficiente para rechazar que los datos tienen esfericidad completa y, por lo tanto, son aptos para el análisis factorial.

También se puede evaluar la esfericidad directamente sobre la matriz mediante la prueba Kaiser-Mayer-Olkin, la cual mide el cuadrado de los elementos de la "imagen" de la matriz comparada con los cuadrados de las correlaciones originales. La imagen consiste en la matriz de covarianzas o correlaciones con el signo opuesto.

En nuestro caso, utilizamos la función `KMO()` del paquete `psych` directamente sobre los datos o sobre la matriz de correlaciones. Esta prueba muestra la Medida de Adecuación del Muestreo (`MSA`. por sus siglas en inglés). Kaiser (1974) describió la interpretación de la `MSA` de la siguiente manera:

$$\begin{aligned} \text{KMO} &\geq 0.9 \Rightarrow \text{Estupendo} \\ \text{KMO} &\geq 0.8 \Rightarrow \text{Meritorio} \\ \text{KMO} &\geq 0.7 \Rightarrow \text{Intermedio} \\ \text{KMO} &\geq 0.6 \Rightarrow \text{Mediocre} \\ \text{KMO} &\geq 0.5 \Rightarrow \text{Miserable} \\ \text{KMO} &< 0.5 \Rightarrow \text{Inaceptable} \end{aligned}$$

Aunque otros usuarios la sugieren de esta forma:

$$\begin{aligned} \text{KMO} &\geq 0.75 \Rightarrow \text{Bien} \\ \text{KMO} &\geq 0.5 \Rightarrow \text{Aceptable} \\ \text{KMO} &< 0.5 \Rightarrow \text{Inaceptable} \end{aligned}$$

Veamos el resultado.

```
kmo = psych::KMO(CL)
kmo
```

```
## Kaiser-Meyer-Olkin factor adequacy
## Call: psych::KMO(r = CL)
## Overall MSA =  0.62
## MSA for each item =
## Matemáticas   Naturales     Francés       Latín
##        0.57        0.65        0.57        0.66
```

Podemos observar que la MSA es *mediocre* para Kaiser y *aceptable* para la mayoría de los usuarios. Digamos que mientras no sea inaceptable, podemos seguir con el análisis.

Además, recordemos que la *imagen* se refiere a la matriz de correlaciones (o covarianzas) con el signo opuesto, es decir, una anti-imagen.

```
# Anti-imagen de correlaciones
kmo$Image
```

```
##             Matemáticas  Naturales    Francés       Latín
## Matemáticas  1.00000000 -0.9184455  0.3625134 -0.04203285
## Naturales   -0.91844547  1.0000000 -0.0735180 -0.29081450
## Francés      0.36251344 -0.0735180  1.0000000 -0.90285267
## Latín       -0.04203285 -0.2908145 -0.9028527  1.00000000
```

```
# Anti-imagen de covarianzas
kmo$ImCov
```

```
##              [,1]         [,2]         [,3]         [,4]
## [1,]  0.047304072 -0.036624361  0.018632688 -0.001800938
## [2,] -0.036624361  0.033615133 -0.003185395 -0.010503746
## [3,]  0.018632688 -0.003185395  0.055847571 -0.042031972
## [4,] -0.001800938 -0.010503746 -0.042031972  0.038808018
```

Notar que la diagonal de la matriz de covarianzas anti-imagen está la estimación de las **unicidades** (la contribución el factor único) de cada variable. Si el modelo factorial elegido es adecuado, los elementos de la diagonal de la matriz de correlaciones anti-imagen deben tener un valor próximo a 1 y el resto de los elementos deben ser pequeños para que las variables compartan factores comunes.

Esto es relevante porque recordemos que el modelo factorial $X_p = a_{ij}F_j + ... + a_{pq}F_q + d_pU_p$ está compuesto por *saturaciones* a_{ij} de la variable X_i en el factor F_j. En el caso de que las variables y los factores estén estandarizados, podemos calcular las **unicidades** d_i^2 dado que

$$var(X_i) = a_{i1}^2 + ... + a_{iq}^2 + d_i^2$$

Para ver nuestro caso directamente, se hace de la siguiente forma:

```
diag(kmo$ImCov)
```

```
## [1] 0.04730407 0.03361513 0.05584757 0.03880802
```

Componentes Principales

Para calcular las componentes principales, existen diversos métodos. Aquí se mostrarán tres: manualmente, con `stats::prcomp()` y con `psych::principal()`.

Manual

Para extraer las saturaciones (a mayor saturación, mayor importancia en la interpretación del eje) y lo que se deriva de ellas, usamos la matriz de correlaciones (o de covarianzas) para conocer los ejes factoriales.

```
# Matriz de correlaciones
mat_cor = cor(CL, use = "pairwise")

# En caso de covarianzas, estandarizamos
mat_cov = cov(CL, use = "pairwise")
dvest = sqrt(diag(mat_cov))
mat_cov = mat_cov / (dvest %o% dvest)

# Comprobamos que ya son idénticas
identical(round(mat_cor, 3), round(mat_cov, 3))
```

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

Luego, obtenemos los valores y vectores propios con la matriz de correlaciones (o de covarianzas) mediante la función `eigen()`. De ella salen

```
autov = eigen(mat_cor)

# Valores propios o autovectores
autovalores = autov$values
autovalores
```

```
## [1] 3.03838967 0.91839441 0.02332132 0.01989460
```

```
# Vectores propios
autovectores = autov$vectors
autovectores
```

```
##            [,1]       [,2]        [,3]        [,4]
## [1,] -0.4635830  0.6072111 -0.18832466  0.61718666
## [2,] -0.5362087  0.3534086 -0.04781922 -0.76504633
## [3,] -0.4578281 -0.6205882 -0.63231919  0.07373043
## [4,] -0.5366233 -0.3482346  0.74994638  0.16837032
```

Con `autovalores` son la variabilidad de las variables latentes. Generalmente se adopta la regla de Kaiser es quedarse con valores propios que sean > 1. Por lo que estrictamente hablando tendríamos que quedarnos con 1 factor. Como el caso es ver si es cierta una clasificación entre materias de ciencias y letras, nos quedaremos con 2 factores, aprovechando que 0.9183944 es cercano a 1. La forma visual para ver esto es a través del `screeplot` o gráfico de sedimentación. Existen múltiples formas de hacerlo (`psych::scree`, `stats::screeplot`), pero en la gráfica 2 se construye con `ggplot2` para mostrar lo que se hace de fondo.

```
ggplot(data.frame(factores = 1:length(autovalores), eig = autovalores),
       aes(x = factores, y = eig)) +
  geom_point(shape = 1) +
  geom_line() +
  geom_hline(yintercept = 1, linetype = "dashed") +
  theme_light() +
  labs(
    x = "factores",
    y = "Autovalor"
  )
```

Además, como los autovalores son la varianza del modelo, podemos calcular su distribución:

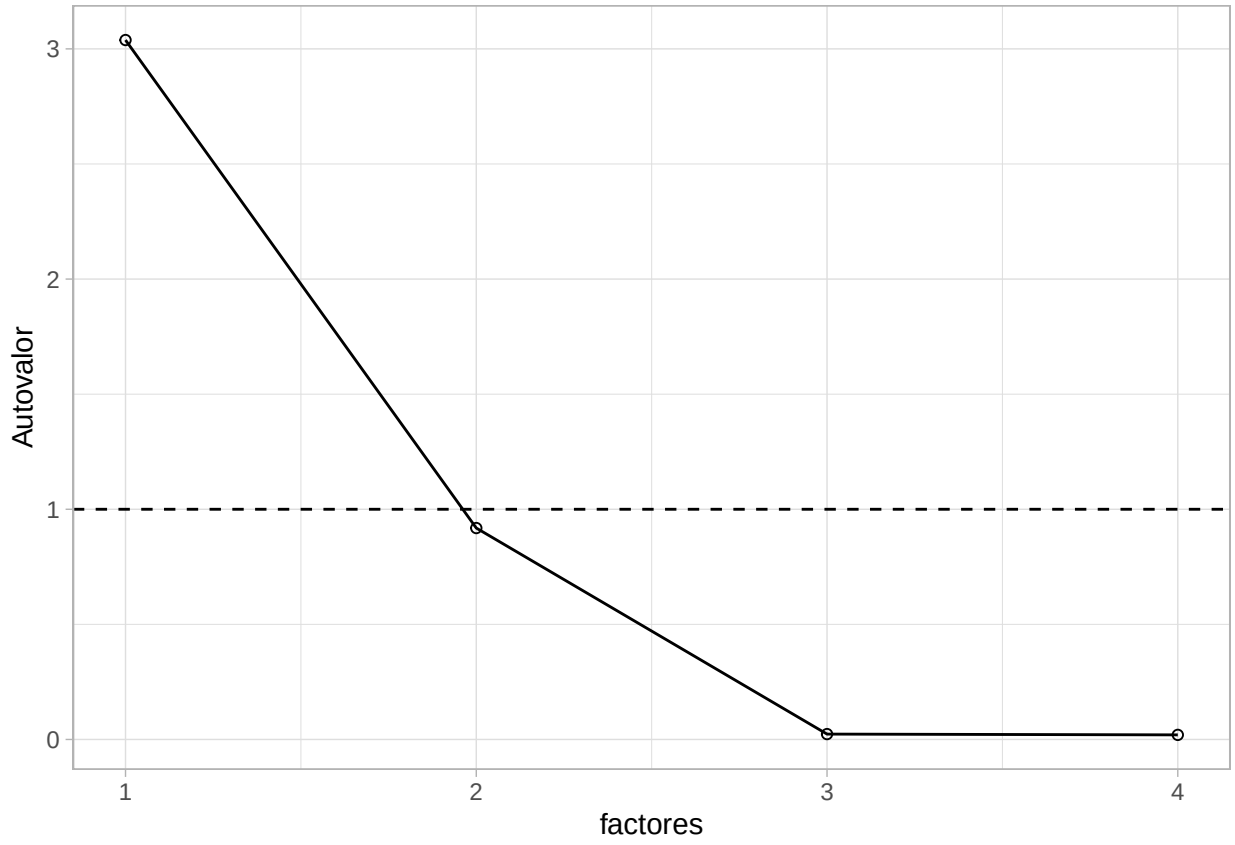

Figura 2: Gráfico de sedimentación

```r
# Proporción de la varianza
prop_varianza = autovalores / length(autovalores)
scales::percent(prop_varianza)
```

```
## [1] "76.0%" "23.0%" "0.6%"  "0.5%"
```

```r
# Varianza acumulada
varianza_acum = cumsum(prop_varianza)
scales::percent(varianza_acum)
```

```
## [1] "76.0%"  "98.9%"  "99.5%"  "100.0%"
```

Vemos que los dos primeros factores explican el 98.9% de la varianza, por lo tanto mantendremos solamente 2 factores. Los seleccionamos de los valores y vectores propios.

```r
autoval = autovalores[1:2]
autovec = autovectores[, 1:2]
```

Para obtener las componentes principales, se asocia el valor propio en orden decreciente con un vector propio. Así, la primera componente será el vector propio con el más alto valor propio. A esto también se le llaman combinaciones lineales y reciben el nombre *cargas* o *loadings* en inglés. Se pueden ver como el peso o la importancia que tiene cada variable en cada una de las componentes.

Al momento de calcular las componentes principales, se busca que las combinaciones lineales sean ortogonales para asegurarse que no estén correlacionadas. En otras palabras, que el producto escalar entre dos componentes sea cero, o geométricamente sean perpendiculares o que el ángulo entre ellas sea de 90º. Esto se hace mediante un proceso iterativo que dura hasta que se calculan todas las posibles componentes o hasta un número determinado menor que el tamaño de todas las variables. Evidentemente, el valor propio asociado al vector propio determinará el orden en que se presentan las componentes.

Para obtener las cargas, multiplicamos matricialmente la matriz diagonal de la raíz cuadrada de los valores propios con la matriz de vectores propios mediante el operador `%*%`:

```r
cargas = autovec %*% sqrt(diag(autoval, nrow = length(autoval)))
colnames(cargas) = c("CP1", "CP2")
rownames(cargas) = colnames(CL)
cargas
```

```
##                    CP1        CP2
## Matemáticas -0.8080704  0.5819080
## Naturales   -0.9346642  0.3386817
## Francés     -0.7980391 -0.5947276
## Latín       -0.9353868 -0.3337233
```

Una vez calculadas las cargas, podemos obtener las comunalidades h_i^2 y las unicidades d_i^2. Las comunalidades son la suma del cuadrado de las cargas de cada ítem, o materias en nuestro ejemplo.

```r
# comunidades
comunalidades = rowSums(cargas^2)
comunalidades
```

```
## Matemáticas   Naturales     Francés       Latín
##   0.9915946   0.9883024   0.9905673   0.9863197
```

Recordemos que las unicidades son $var(X_i) = h_i^2 + d_i^2$. Suponiendo que las variables son reducidas

```r
unicidades = diag(mat_cor) - comunalidades
unicidades
```

```
## Matemáticas   Naturales     Francés       Latín
```

```
## 0.008405358 0.011697558 0.009432656 0.013680349
```

Adicionalmente, podemos comprobar que las sumas de las saturaciones al cuadrado a_i^2 se comportan igual que la varianza explicada:

```
colSums(cargas^2)
```

```
##       CP1       CP2
## 3.0383897 0.9183944
```

Regresando a las cargas, en la gráfica 3 vemos que existe similitud entre la materias que esperaríamos de ciencias y de letras, respectivamente. No obstante, los ejes no muestran alguna información diferenciada. Veamos si rotando la estructura sucede lo mismo.

```
ggplot(
  data.frame(
    Materias = rownames(cargas),
    CP1 = cargas[, 1],
    CP2 = cargas[, 2]
  ),
  aes(CP1, CP2, label = Materias)
) +
  geom_point(color = "blue", size = 3) +
  geom_text(hjust = 0,
            nudge_x = 0.03,
            nudge_y = 0.03) +
  geom_hline(yintercept = 0) +
  geom_vline(xintercept = 0) +
  labs(
    x = paste0(
      "CP1 - ",
      scales::percent(prop_varianza, accuracy = .1)[1],
      " de variación"
    ),
    y = paste0(
      "CP2 - ",
      scales::percent(prop_varianza, accuracy = .1)[2],
      " de variación"
    )
  ) +
  theme_bw() +
  coord_cartesian(xlim = c(-1, 1), ylim = c(-1, 1))
```

Varimax

Aplicamos la rotación "Varimax" de Kaiser con la función `varimax()` sobre las cargas. Lo que hace esta rotación es verificar que la suma de las simplicidades de los factores sea la máxima. La simpliciddad es la varianza de los cuadrados de las saturaciones. Esto lo hace rotando los factores para forzar que unas saturaciones se aproximen a uno y otras a cero, hasta lograr la solución óptima.

```
rot_varimax = varimax(cargas)
cargas_rot = rot_varimax$loadings
cargas_rot
```

```
##
## Loadings:
##             CP1    CP2
```

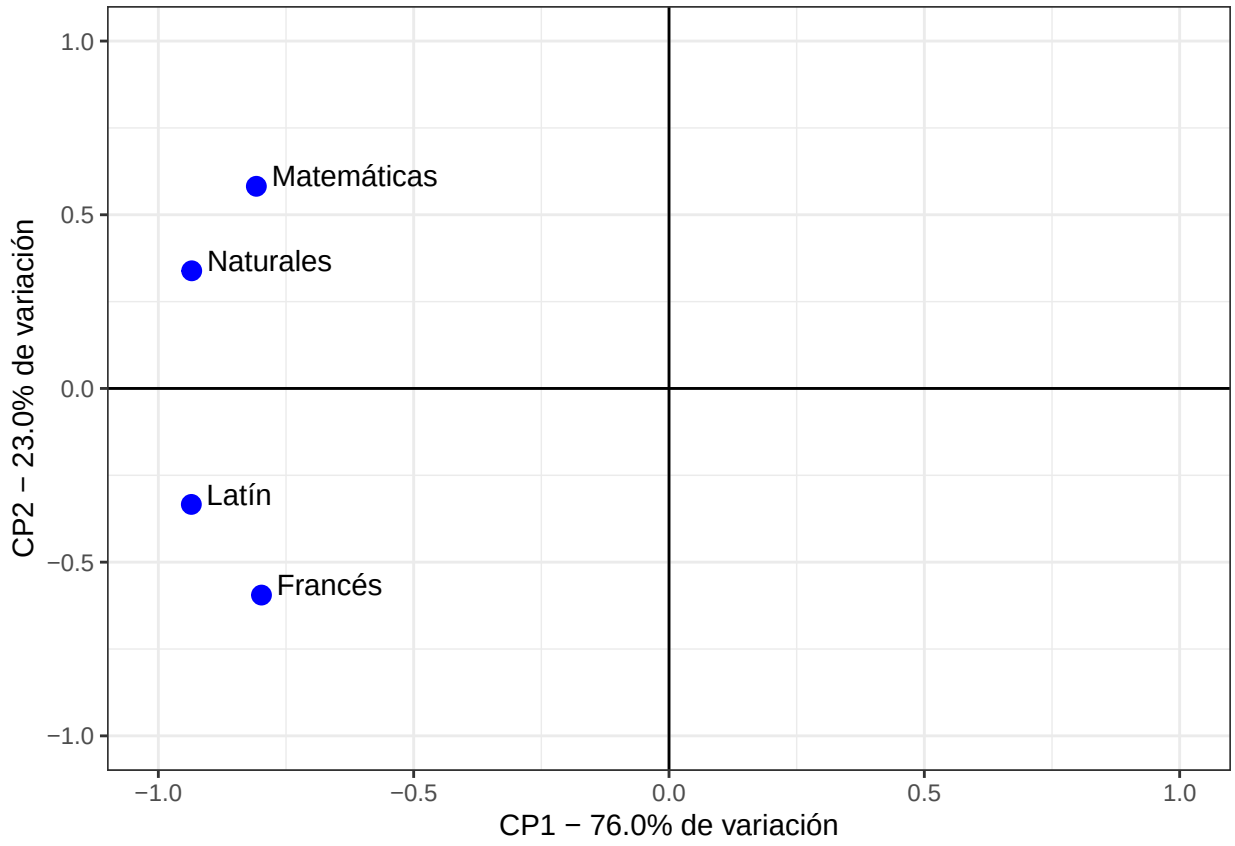

Figura 3: Gráfico de componentes sin rotar

```
## Matemáticas -0.983 -0.156
## Naturales   -0.902 -0.418
## Francés     -0.147 -0.984
## Latín       -0.429 -0.896
##
##                  CP1   CP2
## SS loadings    1.986 1.971
## Proportion Var 0.496 0.493
## Cumulative Var 0.496 0.989
```

Podemos ver que la proporción de varianzas ha cambiado. Además, el valor propio ahora puede ser calculado de la siguiente manera:

```
# Convertir la clase "loadings" a clase "matrix"
cargas_2 = unclass(cargas_rot)

# Obtener la contribución de las variables a cada uno de los factores
colSums(cargas_2^2)
```

```
##      CP1      CP2
## 1.985777 1.971007
```

```
# Proporción de la varianza
prop_var_rot = colSums(cargas_2^2) / 4

# Porcentaje acumulado
cumsum(prop_var_rot)
```

```
##       CP1       CP2
## 0.4964443 0.9891960
```

También existe forma de comprobar que la rotación no alteró las comunalidades:

```
comunalid_rot = rowSums(cargas_2^2)
identical(round(comunalidades, 3), round(comunalid_rot, 3))
```

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

Ahora la solución factorial es más interpretable, diferenciando entre ciencias y letras, con ambos ejes explicando un % de varianza similar.

```
ggplot(
  data.frame(
    Materias = rownames(cargas_2),
    CP1 = cargas_2[, 1],
    CP2 = cargas_2[, 2]
  ),
  aes(CP1, CP2, label = Materias)
) +
  geom_point(color = "green", size = 3) +
  geom_text(vjust = 0,
            nudge_x = 0.05,
            nudge_y = 0.05) +
  geom_hline(yintercept = 0) +
  geom_vline(xintercept = 0) +
  labs(
    x = paste0(
      "CP1 - ",
```

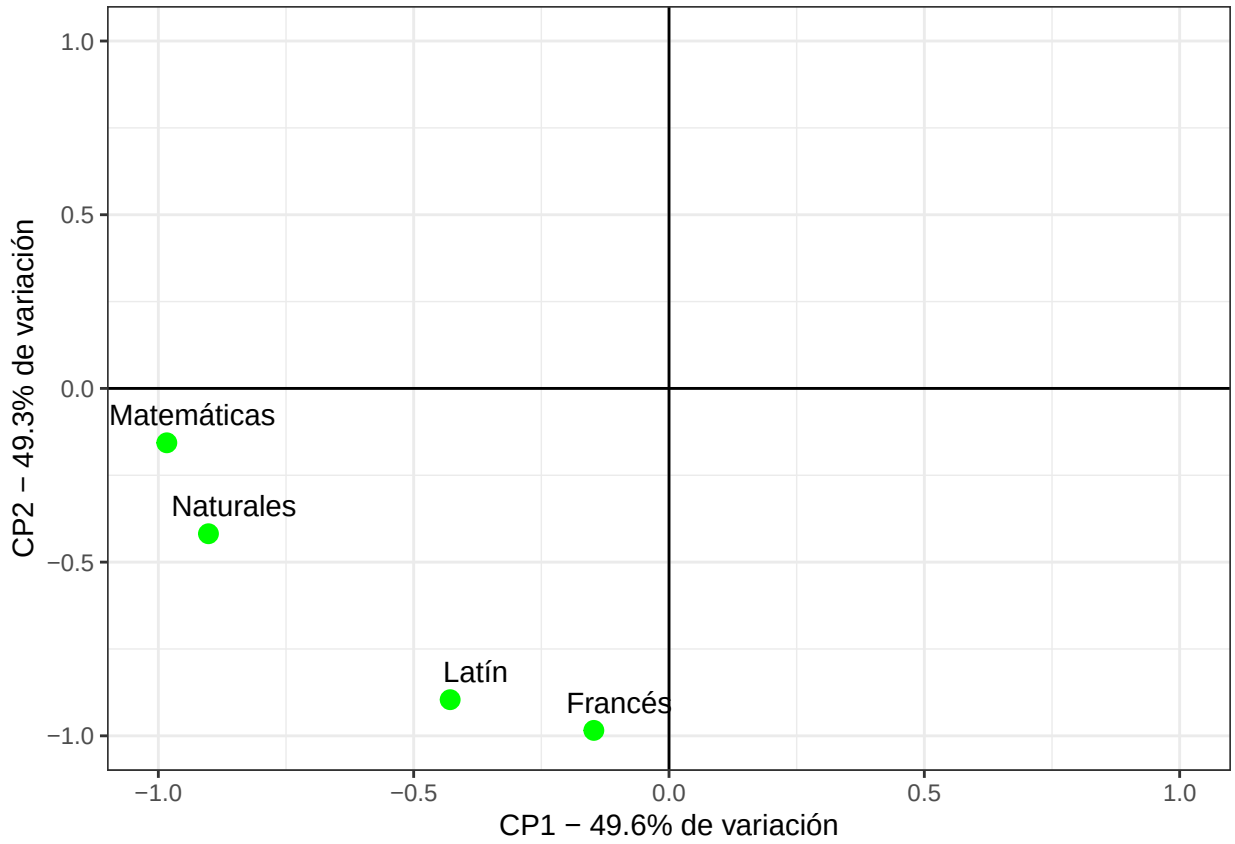

Figura 4: Gráfico de componentes con rotación Varimax

```
      scales::percent(prop_var_rot, accuracy = .1)[1],
      " de variación"
    ),
    y = paste0(
      "CP2 - ",
      scales::percent(prop_var_rot, accuracy = .1)[2],
      " de variación"
    )
  ) +
  theme_bw() +
  coord_cartesian(xlim = c(-1, 1), ylim = c(-1, 1))
```

En la gráfica 4 también podemos corroborar lo mostrado en las cargas rotadas (matriz factorial rotada).

Diferencias con SPSS

Para mostrar las matrices factoriales que muestra SPSS, hay que cambiar el signo de las cargas. Esto se logra de la forma siguiente:

```
# Obtención de signos
signos_totales = sign(colSums(cargas))

# Aplicación sobre matriz factorial sin rotar
cargas_srt = cargas %*% diag(signos_totales)
```

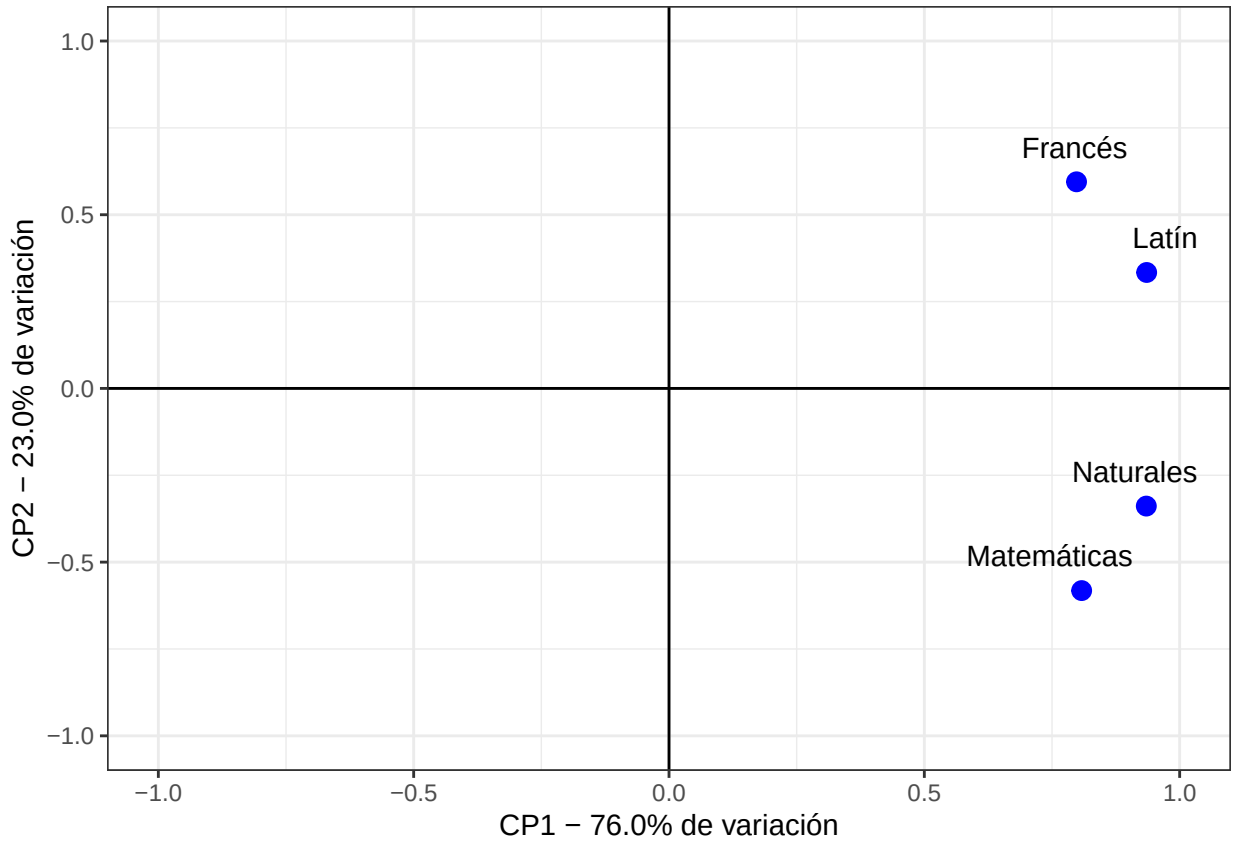

Figura 5: Gráfico de componentes como se muestra en SPSS

```
colnames(cargas_srt) = c("CP1", "CP2")
cargas_srt
```

```
##                   CP1        CP2
## Matemáticas 0.8080704 -0.5819080
## Naturales   0.9346642 -0.3386817
## Francés     0.7980391  0.5947276
## Latín       0.9353868  0.3337233
```

Esto solo cambia la dirección de los vectores propios, pero no altera la varianza:

```
# Varianza explicada o autovalores
colSums(cargas_srt^2)
```

```
##       CP1       CP2
## 3.0383897 0.9183944
```

```
# Prueba de que son idénticas
identical(round(colSums(cargas^2), 3), round(colSums(cargas_srt^2), 3))
```

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

En la gráfica 5 podemos ver este cambio de dirección.

Análogamente, con la rotación *varimax* también se muestran de forma similar a SPSS tanto en la matriz factorial como en la gráfica 6.

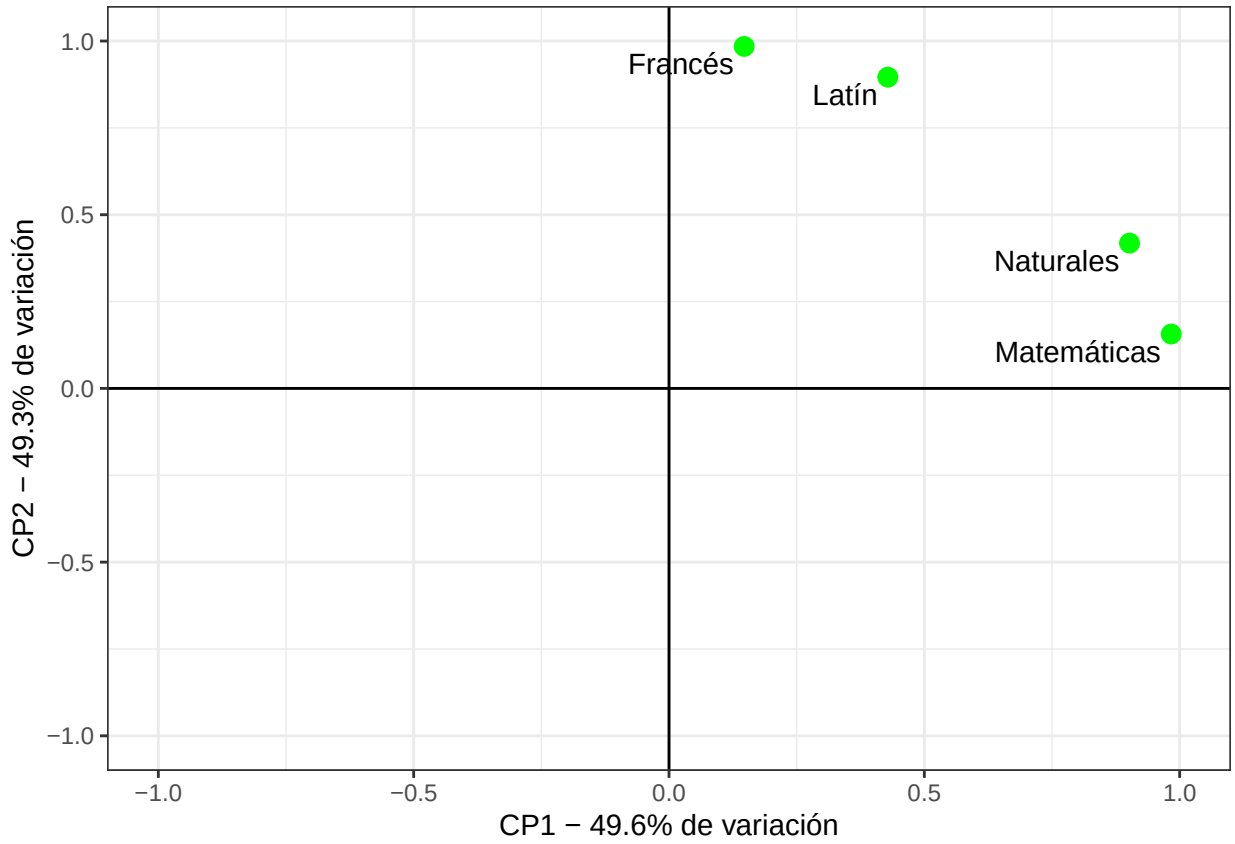

Figura 6: Gráfico de componentes rotados como se muestra en SPSS

```
# matriz de componente rotado
rot_varimax_2 = varimax(cargas_srt)
cargas_rot_2 = rot_varimax_2$loadings
unclass(cargas_rot_2)
```

```
##                    CP1       CP2
## Matemáticas 0.9834143 0.1564962
## Naturales   0.9018541 0.4182841
## Francés     0.1471927 0.9843280
## Latín       0.4285639 0.8959089
```

```
# Varianza explicada
colSums(cargas_rot_2^2)
```

```
##      CP1      CP2
## 1.985777 1.971007
```

Prcomp

La función `prcomp()` devuelve los vectores propios como `rotations` y los valores propios se calculan a partir del cuadrado de `sdev`. Todo lo demás se calcula como la forma manual. Esta función tiene la centralización por defecto, así que solo habrá que escalar para obtener los mismos resultados que el procedimiento anterior.

```
valores_acp = prcomp(CL, scale. = TRUE)
```

```r
# Vectores propios
vec_prop = valores_acp$rotation
vec_prop
```

```
##                    PC1        PC2         PC3         PC4
## Matemáticas -0.4635830 -0.6072111  0.18832466 -0.61718666
## Naturales   -0.5362087 -0.3534086  0.04781922  0.76504633
## Francés     -0.4578281  0.6205882  0.63231919 -0.07373043
## Latín       -0.5366233  0.3482346 -0.74994638 -0.16837032
```

```r
# Valores propios o varianza explicada
val_prop = valores_acp$sdev^2
val_prop
```

```
## [1] 3.03838967 0.91839441 0.02332132 0.01989460
```

```r
# Aplicando el mismo criterio de los factores
cargas_prcomp = vec_prop[, 1:2] %*% diag(valores_acp$sdev[1:2])
cargas_prcomp
```

```
##                   [,1]       [,2]
## Matemáticas -0.8080704 -0.5819080
## Naturales   -0.9346642 -0.3386817
## Francés     -0.7980391  0.5947276
## Latín       -0.9353868  0.3337233
```

```r
# Signo como en SPSS
cargas_prcomp %*% diag(c(-1, -1))
```

```
##                  [,1]       [,2]
## Matemáticas 0.8080704  0.5819080
## Naturales   0.9346642  0.3386817
## Francés     0.7980391 -0.5947276
## Latín       0.9353868 -0.3337233
```

```r
# Porcentaje de varianza explicada
summary(valores_acp)
```

```
## Importance of components:
##                           PC1    PC2     PC3     PC4
## Standard deviation     1.7431 0.9583 0.15271 0.14105
## Proportion of Variance 0.7596 0.2296 0.00583 0.00497
## Cumulative Proportion  0.7596 0.9892 0.99503 1.00000
```

```r
# Comunalidades
rowSums(cargas_prcomp^2)
```

```
## Matemáticas   Naturales     Francés       Latín
##   0.9915946   0.9883024   0.9905673   0.9863197
```

La rotación también se hace con `varimax()` usando `cargas_prcomp`.

```r
varimax(cargas_prcomp)
```

```
## $loadings
##
## Loadings:
##             [,1]   [,2]
## Matemáticas -0.983  0.156
```

```
## Naturales   -0.902  0.418
## Francés     -0.147  0.984
## Latín       -0.429  0.896
##
##                  [,1]  [,2]
## SS loadings    1.986 1.971
## Proportion Var 0.496 0.493
## Cumulative Var 0.496 0.989
##
## $rotmat
##           [,1]       [,2]
## [1,] 0.7095657 -0.7046392
## [2,] 0.7046392  0.7095657
```

Principal

El paquete `psych` ofrece la función `principal`, la cual brinda toda la información que calculamos manualmente. Incluso la matriz factorial es cambiada de signo de la misma forma que SPSS. A diferencia de `prcomp`, la función brinda las cargas con el nombre de `loadings`. Por otra parte, hay que tomar en cuenta que por defecto la función rota los componentes mediante el método *varimax*, por lo que se utiliza el argumento `rotate = "none"` para evitar alguna rotación.

Finalmente, los factores se determinan manualmente, de lo contrario los calcula para todas las variables posibles.

```
acp_psych = psych::principal(CL, nfactors = 2, rotate = "none")

# Cargas y varianza explicada
acp_psych$loadings
```

```
##
## Loadings:
##             PC1    PC2
## Matemáticas  0.808 -0.582
## Naturales    0.935 -0.339
## Francés      0.798  0.595
## Latín        0.935  0.334
##
##                  PC1   PC2
## SS loadings    3.038 0.918
## Proportion Var 0.760 0.230
## Cumulative Var 0.760 0.989
```

```
# Comunalidades
acp_psych$communality
```

```
## Matemáticas   Naturales     Francés       Latín
##   0.9915946   0.9883024   0.9905673   0.9863197
```

```
# Unicidades
acp_psych$uniquenesses
```

```
## Matemáticas   Naturales     Francés       Latín
## 0.008405358 0.011697558 0.009432656 0.013680349
```

Con rotación varimax se ve de la siguiente forma:

```r
acp_psych_rot = psych::principal(CL, nfactors = 2, rotate = "varimax")

acp_psych_rot$loadings
```

```
##
## Loadings:
##             RC1   RC2
## Matemáticas 0.983 0.156
## Naturales   0.902 0.418
## Francés     0.147 0.984
## Latín       0.429 0.896
##
##                  RC1   RC2
## SS loadings    1.986 1.971
## Proportion Var 0.496 0.493
## Cumulative Var 0.496 0.989
```

```r
# residuales de correlación
residuals(acp_psych_rot)
```

```
##             Mtmtc Ntrls Frncs Latín
## Matemáticas  0.01
## Naturales   -0.01  0.01
## Francés      0.00  0.00  0.01
## Latín        0.00  0.00 -0.01  0.01
```

Análisis Factorial

Si en el análisis de componentes principales lo principal era explicar la contribución de las componentes, en el análisis factorial es explicar los factores.

Las estimaciones de las unicidades iniciales que salen de la diagonal principal de la matriz de covarianzas antimagen son las mismas que las que obtuvimos al principio.

```r
diag(kmo$ImCov)
```

```
## [1] 0.04730407 0.03361513 0.05584757 0.03880802
```

La función `psych::fa()` ofrece la versión más completa para el análisis factorial.

```r
AF = psych::fa(CL, 2, rotate = "none", scores = "Anderson")

# Cargas
AF$loadings
```

```
##
## Loadings:
##             MR1    MR2
## Matemáticas  0.806 -0.575
## Naturales    0.932 -0.333
## Francés      0.794  0.586
## Latín        0.932  0.332
##
##                  MR1   MR2
## SS loadings    3.017 0.896
## Proportion Var 0.754 0.224
## Cumulative Var 0.754 0.978
```

```r
# Comunalidades
AF$communality
```

```
## Matemáticas   Naturales     Francés       Latín
##   0.9809225   0.9797095   0.9736405   0.9785692
```

```r
# Unicidades
AF$uniquenesses
```

```
## Matemáticas   Naturales     Francés       Latín
##  0.01907754  0.02029053  0.02635946  0.02143080
```

Notar que los valores propios de la matriz original difieren de las sumas de la extracción de carga al cuadrado (valores propios de la solución del factor común). Esto también se refleja en el porcentaje de la varianza explicada por lo autovalores iniciales contra los extraídos.

```r
# Valores propios de la matriz original
AF$e.values
```

```
## [1] 3.03838967 0.91839441 0.02332132 0.01989460
```

```r
# Proporcion de la varianza
AF$e.values / 4
```

```
## [1] 0.759597417 0.229598602 0.005830329 0.004973651
```

```r
# Suma acumulada de la proporción de la varianza
cumsum(AF$e.values / 4)
```

```
## [1] 0.7595974 0.9891960 0.9950263 1.0000000
```

```r
# Valores propios de la extracción
AF$values
```

```
## [1]  3.016762e+00  8.960793e-01  1.199569e-07 -4.487984e-08
```

```r
# Proporcion de la varianza después de la extracción
AF$Vaccounted
```

```
##                            MR1       MR2
## SS loadings           3.0167623 0.8960793
## Proportion Var        0.7541906 0.2240198
## Cumulative Var        0.7541906 0.9782104
## Proportion Explained  0.7709901 0.2290099
## Cumulative Proportion 0.7709901 1.0000000
```

```r
# residuales de correlación
residuals(AF)
```

```
##             Mtmtc Ntrls Frncs Latín
## Matemáticas 0.02
## Naturales   0.00  0.02
## Francés     0.00  0.00  0.03
## Latín       0.00  0.00  0.00  0.02
```

En suma, la solución no es interpretable. A pesar de ello, el modelo con dos factores comunes y las correspondientes unicidades es bueno (mejor que el de componentes principales) ya que los residuos son todavía más pequeños.

Ahora se aplica la rotación varimax para capturar las unicidades de los factores.

```r
AF_rot = psych::fa(CL, 2, rotate = "varimax", scores = "Anderson")

AF_rot$loadings
```

```
##
## Loadings:
##             MR1   MR2
## Matemáticas 0.978 0.154
## Naturales   0.899 0.415
## Francés     0.155 0.974
## Latín       0.432 0.890
##
##                  MR1   MR2
## SS loadings    1.976 1.937
## Proportion Var 0.494 0.484
## Cumulative Var 0.494 0.978
```

```r
# Comunaliddades
AF_rot$communality
```

```
## Matemáticas   Naturales     Francés       Latín
##   0.9809225   0.9797095   0.9736405   0.9785692
```

```r
#
AF_rot$Vaccounted
```

```
##                            MR1       MR2
## SS loadings           1.9757483 1.9370934
## Proportion Var        0.4939371 0.4842733
## Cumulative Var        0.4939371 0.9782104
## Proportion Explained  0.5049395 0.4950605
## Cumulative Proportion 0.5049395 1.0000000
```

```r
# residuales de correlación rotada
residuals(AF_rot)
```

```
##             Mtmtc Ntrls Frncs Latín
## Matemáticas 0.02
## Naturales   0.00  0.02
## Francés     0.00  0.00  0.03
## Latín       0.00  0.00  0.00  0.02
```

El modelo mejora la distribución del % de la varianza explicada entre los factores. Notar que en este caso el % de varianza no es idéntico al inicial, ni en la solución no rotada ni en la rotada. La estimación inicial de las comunalidades son los coeficientes de correlación múltiple. Tanto en la solución no rotada como en la rotada, cambian las saturaciones y por tanto las contribuciones de los factores comunes a las variables y las contribuciones de las variables a los dos primeros ejes, valores sobre los cuales se recalcula la absorción de inercia.

Por otra parte, los residuos muy bajos ponen en manifiesto que el modelo fue adecuado.

Finalmente, vemos los resultados de la rotación en la gráfica **??**.

Los resultados son similares a los obtenidos con componentes principales, pero al tener en cuenta la unicidad, hemos conseguido que los dos factores comunes tengan importancia comparable al capturar cada uno la mitad de la varianza.

Kaiser, Henry F. 1974. «An index of factorial simplicity». *Psychometrika* 39 (1): 31-36. https://doi.org/10.1007/BF02291575.