

Trabajo de Final de Grado

MARTEST

Hacia una evaluación hecha a medida.

Autor Carlos Pérez
Tutora Carme Viladrich

Grado en Psicología
Universidad Autònoma de Barcelona

19 de abril de 2021

Índice general

Índice general	II
1 Respuestas en blanco	1
1.1 Introducción a los valores en blanco	1
Mecanismos	1
Patrones	2
Influx y Outflux	3
1.2 Métodos	4
Imputación	4
2 Teoría de Respuesta al Ítem	6
2.1 Teoría de Respuesta al Ítem Unidimensional	7
Principios	7
Modelos	7
2.2 Teoría de Respuesta al Ítem Multidimensional	10
Modelos	10
3 Obtención de parámetros	14
Introducción al Análisis Factorial	14
3.1 Modelo exploratorio	15
3.2 Modelo confirmatorio	17
Modelos two-tier	18
3.3 Bondad de ajuste	19
Principio de monotonicidad	19
Análisis de Varianza (ANOVA)	20
Convergencia del modelo	21
itemfit	22
M2	22
residuals	23
clusters	24
4 Test Adaptativo Informatizado	26
4.1 mirtCAT	27
Resultados	30
4.2 Consideraciones	33
4.3 Ejemplo	33
Bibliografía	36

Apéndice	39
1 Apéndice A: Pequeño tutorial en R	39
Instalar R y RStudio	39
Instalar y cargar paquetes	40
Documentación y guardado	40
Datos	41
Estructuras de datos	42
Indexar	43
Funciones	44
2 Apéndice B: Funcionamiento del ítem	45
3 Apéndice C: Optimización	47
Estimación de habilidad	48
4 Apéndice D: Seleccionar el próximo ítem del banco de ítems	50
Maximizar la información en la dirección de menor información	50
5 Apéndice E: Código	51
6 Apéndice F: Funciones	53
patrones	53
plotflux	53
polar_coords	54

Respuestas en blanco

1

1.1. Introducción a los valores en blanco

Los valores en blanco en **psicometría** son un problema constante que llevan a la estimación de parámetros sesgados o a la pérdida de potencia estadística (Roth, 1994). La potencia hace referencia a la capacidad que tiene una prueba estadística para descubrir las relaciones en una base de datos y el sesgo, en cambio, produce que dos grupos de personas con el mismo nivel de habilidad rindan de forma diferente en una prueba (Abad y col., 2011). En el ámbito académico un dato faltante se traduce en dejar ítems o preguntas en blanco, algo muy común hoy en día, ya que no se suele obligar a las personas examinadas a contestar todos los enunciados de un examen.

Con el avance de los ordenadores se han creado y mejorado métodos para paliar la falta de información, pero estos no están exentos de sesgo en las estimaciones o de alteraciones en las relaciones entre variables y en las varianzas, etc. Además, estos métodos no solo se ven afectados por el porcentaje de valores en blanco, también son importantes los patrones y los mecanismos que explican la falta de datos.

Mecanismos

Des de Rubin (1976) se definieron tres posibles causas que pueden explicar los datos en blanco:

- **MCAR** *Missing Completely At Random*

Los datos en blanco no tienen ninguna causa sistemática, es decir, son completamente aleatorios; p. ej., se daría cuando una persona contesta un examen sin leer ningún enunciado.

- **MAR** *Missing At Random*

Sucede cuando el valor en blanco depende de una característica o variable medible que pueda ser observada, es decir, de una variable de la cual se tengan datos; p. ej., el tiempo de estudio de cada alumno o alumna determina su nota, por lo tanto, si se conociese, se tendrían un mecanismo MAR.¹

- **MNAR** *Missing Not At Random*

Significa que un valor en blanco proviene de una variable desconocida; p. ej., se daría cuando no se sabe si una persona copia o no.

La principal consecuencia de datos **MCAR** es la pérdida de potencia estadística, aunque los análisis no presenten sesgos; lo mismo se aplica a **MAR** si se conoce la causa de la pérdida pero **MNAR** es considerado un problema porque siempre da resultados inestables en la estimación

1.1 Introducción a los valores en blanco	1
Mecanismos	1
Patrones	2
Influx y Outflux	3
1.2 Métodos	4
Imputación	4

Psicometría

La psicometría se define como aquella rama de la Psicología Experimental encargada de la medición y la cuantificación de los procesos psicológicos de la psique humana; en ella se encuentra la construcción de tests, cuyas propiedades pueden ser estudiadas por la Teoría Clásica de los Tests (TCT) o bien la Teoría de Respuesta al Ítem (TRI) y tienen como objetivo estudiar las puntuaciones observadas, medir un valor inobservable del dominio o rasgo psicológico a evaluar y estimar el error de medición.

1: Por ahora no hay ningún método para detectar un mecanismo **MNAR**, para **MAR** tampoco, aunque presenta más facilidades que **MNAR** (Little & Rubin, 2002); **MCAR** es detectable con el test de Little (Little, 1988), pero es controvertido e inverosímil que todas las personas contesten aleatoriamente a un examen y el test es inadecuado en algunos casos (Rhoads, 2012).

de parámetros, a menos que se especifique un modelo que explique su mecanismo, cosa que es francamente difícil (Allison, 2014).

Patrones

Como se ha visto hasta ahora, las consecuencias de los datos faltantes se pueden explicar por la causa de estos (MCAR, MAR, MNAR), es decir, ¿por qué faltan datos? pero también se pueden explicar por los patrones observados o dicho de otro modo, por aquello que muestran los datos (Rönkkö, 2020).

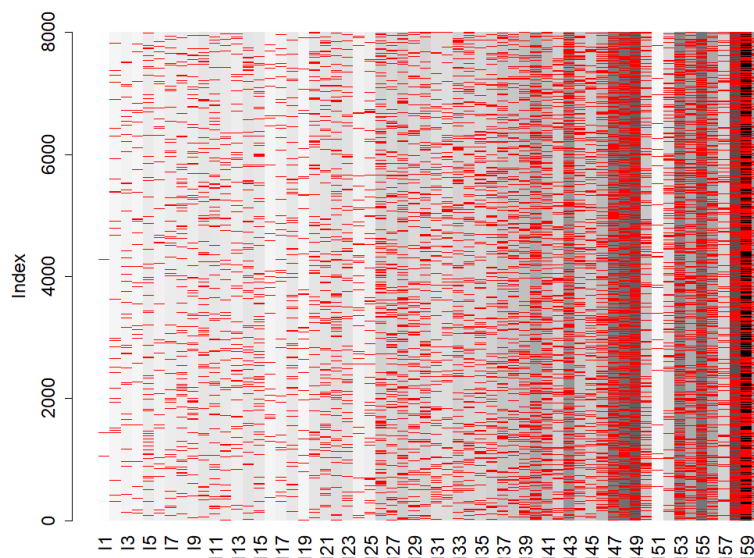
Estos patrones pueden informar de qué mecanismo hay detrás, además de informar de qué ítems pueden conllevar problemas en la imputación. Con ellos se pueden obtener estadísticos que muestran la relación e influencia entre pares de ítems y entre el ítem y la resta de datos.

La función *patrones* crea un gráfico que permite observar qué patrón existe.

```
1 patrones(df)
```

Figura 1.2

Gráfico de la función *patrones*

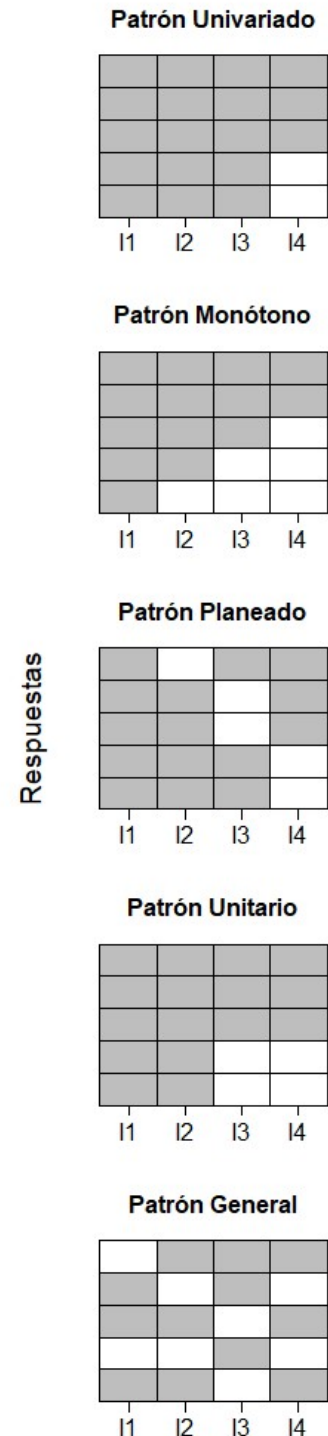


Nota. El gráfico muestra como las respuestas en blanco (rojo) se distribuyen a lo largo de los 60 ítems y las 8000 personas, el color de cada columna informa del porcentaje de valores faltantes; cuánto más negros, más alto el porcentaje.

La Figura 1.2 demuestra que se estaría delante de un patrón monótono, ya que, a medida que avanza el test se observan más valores en blanco. Si se supiese por qué se da este patrón estaríamos delante de un mecanismo MAR y se podrían hacer mejores imputaciones, ya que no se observa un mecanismo MCAR.

Figura 1.1

Tipos de patrones.



Nota. Respuestas de cinco personas a 4 ítems; en gris los contestados. Imagen reproducida de *Flexible Imputation of Missing Data* (p.4), S. van Buuren, 2018, Routledge, <https://stefvanbuuren.name/fimd/>, CC BY 4.0.

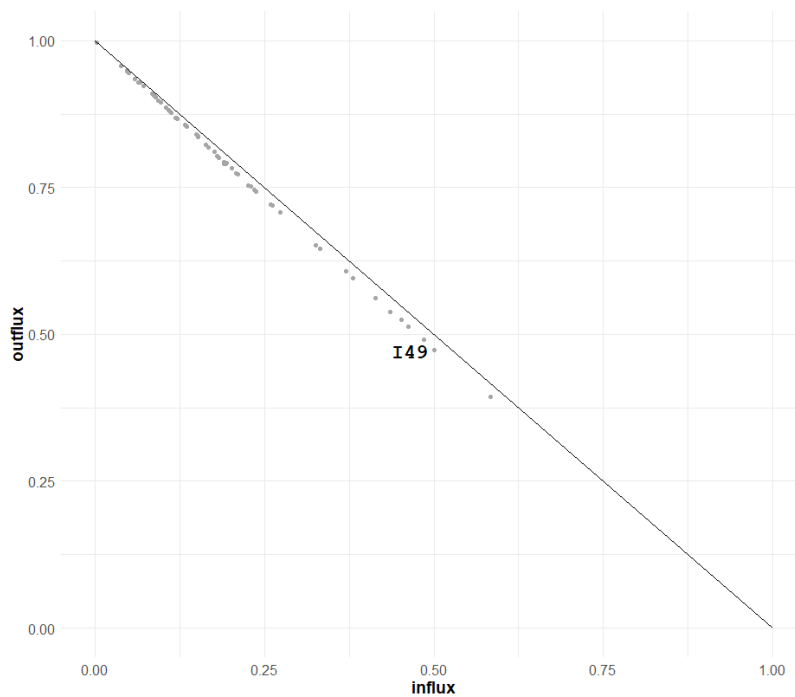
Influx y Outflux

Los estadísticos no basados en pares de ítems son el **Influx**, el cual informa de cómo se relaciona el ítem con los datos observados (es análogo a la proporción de datos en blanco del ítem y los valores cercanos a 1 indican mejor imputabilidad); y el **Outflux**, que informa de como el ítem contribuye a la imputación de los demás.*

2 `plotflux(df)`

Figura 1.3

Influx y Outflux



Nota. El gráfico muestra el **Influx** y **Outflux** de cada ítem. Aquellos que tienen ambos índices bajos por debajo de 0.5 son marcados en la gráfica. La etiqueta hace referencia al nombre del ítem que tiene la base de datos introducida.

Aunque no haya un punto de corte para Influx o Outflux en van Buuren (2018), `plotflux` muestra unos valores equilibrados que no tienen por qué significar una mala imputación.

* Para más información consultar (van Buuren, 2018)

1.2. Métodos

Los métodos para lidiar con datos omitidos se pueden clasificar según si hacen una delección parcial, como la delección listwise o la pairwise o si hacen una imputación de estos. Los dos métodos, tanto la imputación como la delección, buscan tener datos completos, pero la imputación es, generalmente, la primera opción (Sulis & Porcu, 2017) porque permite aplicar métodos con los datos completos sin perder potencia estadística, como sí harían la delección listwise y pairwise.

Imputación

Hay muchos métodos de imputación, ya sean los de máxima verosimilitud, sustitución por media o mediana y procedimientos Hot-Deck, entre otros, y estos se podrían clasificar en métodos de imputación múltiple o univariada, los primeros imputan datos plausibles y crean varias simulaciones y los segundos solo crean una simulación (Graham, 2009). Aunque la imputación múltiple tiene en cuenta la incerteza de los valores reales desconocidos, requiere aplicar un procedimiento estadístico, como una media aritmética, a cada muestra simulada; para tener una eficiencia casi perfecta se necesitan hacer infinitas simulaciones (Allison, 2014); esto es incompatible con procedimientos que se verán a lo largo del manual, ya que necesitan resultados deterministas. **

Por otro lado, entre los métodos univariados se encuentra el algoritmo *esperanza-maximización* o algoritmo **EM**. Este procedimiento no está solo destinado a la imputación de datos, en estadística tiene muchas aplicaciones relacionadas con algoritmos de clasificación o agrupación, ya que estima las propiedades de algo que se desconoce, en este caso, respuestas a un examen («Algoritmo esperanza-maximización», 2020).

Al ser un algoritmo, este se basa en un procedimiento iterativo, que consta de dos fases y que aplicado a la imputación de valores se resumen en:

1. Imputar cada valor perdido utilizando un modelo predictivo que tome como predictores las demás variables, p. ej. hacer una regresión logística con los demás ítems para imputar las respuestas de uno solo.²
2. Actualizar los parámetros varianza, covarianza y media de cada ítem, a partir de los datos imputados.

2: El primer ítem que se imputa sirve de predictor para los demás y aún tiene datos en blanco. Esas nuevas respuestas añaden un error de predicción que, con cada iteración, irá disminuyendo.

Cuando los cambios de iteración a iteración en los parámetros son tan pequeños como para considerarse triviales, se dice que el algoritmo ha convergido y se obtiene una solución satisfactoria (Graham, 2009).

** Para implementar imputación múltiple en **R** consultar el paquete *mice* (van Buuren & Groothuis-Oudshoorn, 2011) y el manual de van Buuren (2018).

En R, el algoritmo EM se puede implementar con la función *Amelia()* (Honaker y col., 2011).

```
3 df <- amelia(df, noms = 1:60, m = 1)$imputations$imp1

-- Imputation 1 --

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
21 22 23
```

La función muestra el número de iteraciones del algoritmo hasta que converge; en este caso 23 iteraciones. Es posible que 'Amelia()' dé el siguiente error:

```
Amelia Error Code: 43
You have a variable in your dataset that does not vary. Please remove this variable.
Variables that do not vary: I1
```

Esto significa que el ítem 'I1' no aporta ningún tipo de información, ya que puede haberse contestado erróneamente o correctamente por todas las personas y, por lo tanto, debe eliminarse:

```
4 NombreBaseDeDatos <- NombreBaseDeDatos[, -1]

plotflux() también puede informar de ello, ya que, si hubiese un ítem no imputable, resaltaría en rojo su nombre dentro de la gráfica. La base de datos inicial (df) se sobrescribe por la matriz de datos completa y, a partir de ahora, se usará esta durante todo el manual3.
```

La función toma como primer argumento la matriz de datos; en el segundo, se especifica que todas las columnas tiene valores nominales, ya que representan respuestas binarias; tercero, se elige una sola imputación, ya que no se quiere hacer imputación múltiple y cuarto, se extra la base de datos con las respuestas imputadas a través de *\$imputations\$imp1*.

3: Aunque aquí se sobrescriba el nombre para evitar confusiones es recomendable no hacerlo para no perder el objeto que contenía la base de datos inicial y tener un mayor orden en el entorno de trabajo.

Con preguntas de respuesta binaria – de sí o no – sería lógico pensar que una persona tiene o no tiene el conocimiento suficiente para contestar correctamente, a diferencia de una respuesta abierta como – *¿Qué eventos históricos ocurrieron en el siglo XVII?* –. Además, para ciertas preguntas, cada alumno o alumna tiene una estrategia diferente para contestar, p. ej., – *¿Cuál es el número áureo?* – se podría contestar de memoria o demostrando $\frac{1+\sqrt{5}}{2}$.

Aunque estos sucesos podrían ser discutibles en algunos casos, en este manual se va a asumir que el nivel de una persona es algo continuo y que se necesita más de una estrategia¹ para contestar a una pregunta.

Para un test adaptativo, se necesita representar la interacción de un ítem con una persona, es decir, establecer cuántas dimensiones evalúa, cómo las evalúa y qué nivel de rasgo y probabilidad tiene cada persona de contestarlo correcta o incorrectamente (Oppl y col., 2017). Para estas premisas, en psicometría se definieron modelos matemáticos que conseguían definir esa interacción en el primer cuarto del siglo XX y que posteriormente sentarían las bases de la Teoría de Respuesta al Ítem Unidimensional (TRI) (Reckase, 2009).

Aunque la TRI solo evalúe una dimensión, se explicarán sus bases a continuación para presentar los conceptos básicos sobre la interacción persona-ítem con modelos matemáticos más simples que los que se verán más adelante en la Teoría de Respuesta al Ítem Multidimensional (MIRT *en inglés*). Para ello se tratarán conceptos complejos para alguien que no sepa probabilidad o álgebra, de una forma visual y accesible; aunque es aconsejable saber las bases de la TRI y la MIRT, si se quiere pasar directamente a su aplicación vaya al capítulo Obtención de Parámetros.

Ya que este manual no está destinado a ítems de respuesta múltiple, no se tratarán los Modelos de Respuesta Graduada (MRG) tanto para la TRI como para la MIRT. Para más información sobre otros modelos y aspectos teóricos más allá de los explicados a continuación consulte manuales como el de van der Linden (2019) para la TRI y Reckase (2009) para la MIRT. Además, en el Apéndice B se puede encontrar una explicación más detallada de cómo los modelos de la MIRT, que se van a explicar más adelante, afectan al funcionamiento del ítem.

2.1 Teoría de Respuesta al Ítem Unidimensional	7
Principios	7
Modelos	7
2.2 Teoría de Respuesta al Ítem Multidimensional	10
Modelos	10

1: A partir de ahora, las diferentes estrategias se van a llamar dimensiones, rasgos o *theta* (θ).

2.1. Teoría de Respuesta al Ítem Unidimensional

Principios

Por ahora, se han comentado algunos principios de la interacción persona-ítem, pero los modelos matemáticos, tanto de la TRI como de la MIRT, necesitan varias asunciones que deben cumplirse:

- **El nivel de rasgo no puede verse afectado durante el test.**
Este principio se vulnera cuando se aprenden cosas nuevas durante el examen debido a la interacción con los ítems. Por lo tanto, el nivel de rasgo de una persona debe ser constante durante toda la prueba.
- **Las características de un ítem se mantienen constantes independientemente de a quién se le aplique.**
Aunque una ítem evalúe una o más dimensiones, estas no pueden variar en número dependiendo de la persona que lo conteste.
- **Las respuestas a un ítem son independientes a las respuestas de otro ítem.**
Este principio, llamado independencia local, establece que, p. ej., el enunciado de una pregunta no puede ayudar a responder otra.
- **El rendimiento en los ítems solo depende del nivel de la persona en un único rasgo, es decir, el test debe ser unidimensional.**²
- **A medida que el nivel de rasgo aumenta, la probabilidad de acertar el ítem también debe aumentar.**
A esta premisa se le llama principio de monotonicidad.

2: Para la TRI, este principio es la diferencia fundamental entre la MIRT, cuyos principios son idénticos, excepto en el de unidimensionalidad.

Modelos

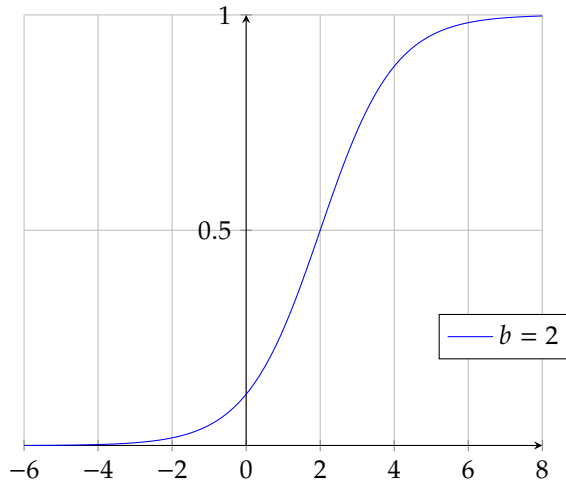
Rasch (1960) propuso un modelo que cumplía estos principios, introduciendo de esta forma la función logística de probabilidad acumulada como base de los modelos de la TRI.

$$P_{ij}(\theta_j, b_i) = \frac{e^{\theta_j - b_i}}{1 + e^{\theta_j - b_i}} \quad (2.1)$$

$P_{ij}(\theta_j, b_i)$ significa que la probabilidad P de acertar un ítem i por una persona j depende del nivel θ de la persona j y la dificultad b del ítem i .

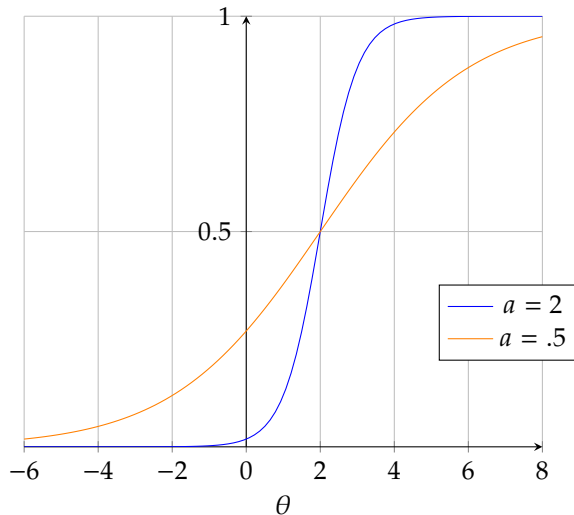
Esta función, como se ve en la Figura 2.1, determina la probabilidad de acertar un ítem para cualquier nivel de θ ³ y contiene solo dos parámetros: θ que corresponde al nivel de rasgo de la persona y b que equivale a la dificultad del ítem; a esta curva se le llama Curva Característica del Ítem (CCI). Una persona que tiene el mismo nivel de θ que la dificultad del ítem tendrá un 50 % de probabilidades de acertarlo.

3: Ahora el nivel de rasgo no solo es un continuo, sino que este va de $-\infty$ a ∞ .

Figura 2.1*Modelo de un parámetro (1 PL) o de Rasch*

Nota. El gráfico muestra como es el comportamiento de la función logística. Esta es exponencial y su crecimiento decrece a medida que llega a un límite y tanto su extremo inferior como superior son asintóticos, es decir, jamás llegan a 0 o 1. Nótese también, que b equivale a qué nivel de θ se da el punto de inflexión de la curva.

En 1968, Birnbaum añadió el parámetro de discriminación a al modelo de Rasch, que determina que pendiente tiene la CCI, pero a nivel psicométrico, a ⁴ determina cómo el ítem es capaz de diferenciar entre valores de θ que pueden o no contestar correctamente el ítem.

Figura 2.2*Modelo de dos parámetro (2 PL)*

Nota. Se observa como un ítem con $a = 2$ tiene una curva más vertical que uno con $a = 0.5$ y, por lo tanto, es capaz de determinar con más precisión qué niveles de θ lo contestarán correcta o incorrectamente.

Los modelos para ítems de dos categorías normalmente se nombran según el número de parámetros que tienen; en este caso solo contiene b . El de un parámetro también se llama modelo de Rasch.

4:

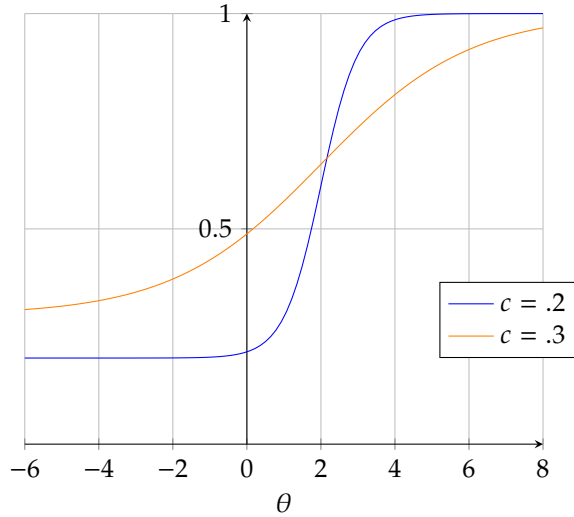
$$P_{ij}(\theta_j, b_i, a_i) = \frac{e^{a_i(\theta_j - b_i)}}{1 + e^{a_i(\theta_j - b_i)}} \quad (2.2)$$

Aunque se consiga representar el nivel de habilidad de una persona en un continuo que va de $-\infty$ a ∞ , al fin y al cabo, es preferible tener ítems que establezcan puntos de corte bien definidos, dicho de otro modo, que tengan buena discriminación.

En 1980/2012 Lord incluyó el parámetro c^5 o parámetro de pseudoazar, que se utiliza para representar la probabilidad de contestar una respuesta al azar.

Figura 2.3

Modelo de tres parámetro (3 PL)

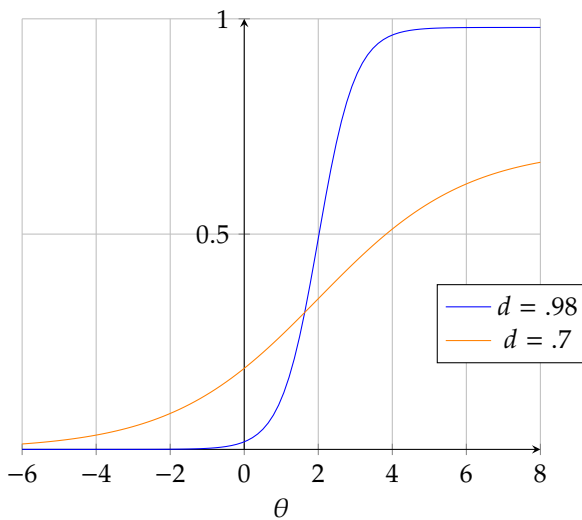


Nota. Se puede ver como se da una probabilidad de acierto igual a c a cualquier persona con un nivel bajo de θ .

Finalmente, en 1981, Barton y Lord establecieron el modelo de 4 parámetros (4PL), que introduce el parámetro d^6 de inatención, el cual define la altura máxima de la asíntota superior. Este parámetro sirve para no penalizar tanto a las personas con altos niveles de θ que puedan haber fallado durante el test por una falta de atención.

Figura 2.4

Modelo de cuatro parámetro (4 PL)



5:

$$P_{ij}(\theta_j, b_i, a_i, c_i) = c_i + (1 - c_i) \frac{e^{a_i(\theta_j - b_i)}}{1 + e^{a_i(\theta_j - b_i)}} \quad (2.3)$$

Parecería lógico utilizar siempre un modelo de 3 parámetros, ya que cualquier ítem puede ser respondido al azar, pero se utiliza principalmente para ítems de respuesta múltiple y también requiere muestras más grandes que los otros porque tiene más parámetros a estimar (Harris, 1989).

6:

$$P_{ij}(\theta_j, b_i, a_i, c_i, d_i) = c_i + (d_i - c_i) \frac{e^{a_i(\theta_j - b_i)}}{1 + e^{a_i(\theta_j - b_i)}} \quad (2.4)$$

A nivel estadístico una asíntota superior de 0.98 o 0.99 sí que tiene utilidad porque permite estimaciones más precisas del nivel de habilidad en un TAI (Liao y col., 2012).

2.2. Teoría de Respuesta al Ítem Multidimensional

Como se ha explicado al comienzo de este capítulo, un ítem puede evaluar más de una habilidad o rasgo a la vez, por lo tanto, a diferencia de la TRI ahora se tendrá más de un valor de θ para cada persona; la probabilidad de acertar un ítem vendrá determinada por el valor de θ que se tenga en cada dimensión. Otro aspecto importante, como se verá más adelante, es que se debe distinguir entre las dimensiones que define la MIRT (θ) con los constructos que representan un aspecto cognitivo (aritmética, comprensión lectora, etc); es posible que un modelo establezca dos dimensiones para un test con tres clusters (Reckase, 2009).

Modelos

Todos los modelos se dividen en dos grandes grupos y sus diferencias radican en la forma en la que los valores de θ , junto con las características de cada ítem, determinan la probabilidad de acierto al ítem.

El primer grupo asume que los valores de θ son una combinación lineal de parámetros, es decir, que la capacidad de acertar un ítem es la suma de varios conocimientos al mismo tiempo, lo que implica que distintos niveles de rasgo en varias dimensiones puedan dar la misma probabilidad de acierto.⁷

A estos modelos se les llama compensatorios y tiene una visión holística de como alguien, junto a su nivel de θ en cada una de las dimensiones, puede acertar un ítem.

El segundo grupo de modelos asume que la respuesta a un ítem depende del nivel de θ en cada una de las dimensiones, haciendo que la probabilidad de un ítem sea el producto de acertar ese ítem en cada una de las dimensiones. Este modelo asume que para acertar un ítem se deben tener altos niveles de pericia en cada dimensión y por eso se llaman modelos no compensatorios, aunque también se pueden llamar parcialmente compensatorios.⁸

En la literatura actual los modelos más extendidos son los compensatorios tanto para ítems dicotómicos como politómicos, siendo los primero los más estudiados. Algunas contextos educativos requieran un modelo en vez del otro por sus hipótesis sobre cómo se da la interacción entre respondiente e ítem. En la práctica pocos estudios se han hecho sobre qué modelo es mejor, aunque Bolt y Lall (2003) y Spray y col. (1990) sugirieron que los valores de θ suelen funcionar de una forma compensatoria.

Cluster

APA (American Psychological Association, s.f.) define clustered data como:

a set of observations or scores that can be grouped into multiple subsets (clusters), such that the items in each subset are similar to one another with respect to certain attributes and the distinctions between subsets help explain the overall variation among the values as a whole. [un conjunto de observaciones o puntuaciones que pueden agruparse en múltiples subconjuntos (clusters), de forma que los elementos de cada subconjunto son similares entre sí con respecto a ciertos atributos y las distinciones entre los subconjuntos ayudan a explicar la variación global entre los valores en su conjunto.]

7: En el ejemplo introductorio a este capítulo se proponían dos formas de responder correctamente a – ¿Cual es el número áureo? –, para ello se pueden desarrollar dos estrategias que no son mutuamente excluyentes, por lo tanto, dos personas que tengan diferentes niveles en memoria o cálculo pueden tener la misma probabilidad de acierto.

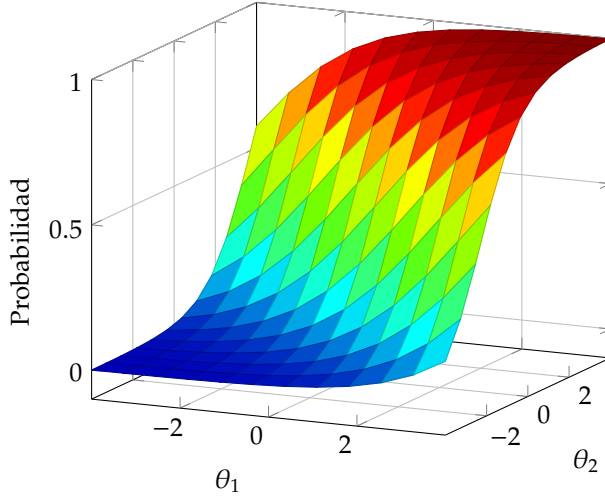
8: Es habitual llamarlos no compensatorios pero Reckase (2009) los nombra parcialmente compensatorios, ya que niveles de θ altos en una sola dimensión dan una probabilidad de acierto mayor que los bajos, por lo tanto, algún tipo de compensación ocurre.

Modelos compensatorios basados en la TRI

En los modelos de la TRI, la curva definida por la función logística era la Curva Característica del Ítem, pero aquí, ya no se puede hablar de una curva en 2 dimensiones: en la MIRT se habla de la Superficie Característica del Ítem (SCI)(Reckase, 2009), como se ve en la siguiente figura.

Figura 2.5

SCI del modelo de dos parámetros



Nota. SCI de un ítem con $a_1 = .75$, $a_2 = 1$ y $d = -.7$. Los modelos compensatorios cumplen el principio de monotonicidad, ya que a medida que aumenta θ_1 y θ_2 también lo hace la probabilidad de acierto.

La SCI se define por

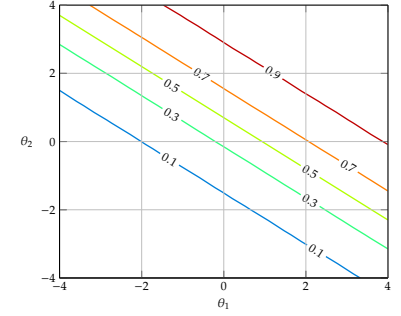
$$P_{ij}(\theta_j, a_i, d_i) = \frac{e^{a_i \theta'_j + d_i}}{1 + e^{a_i \theta'_j + d_i}} \quad (2.5)$$

La Figura 2.5.1 muestra un gráfico de contorno donde se ve como varias combinaciones de θ tienen la misma probabilidad de acierto; se observa que niveles bajos de θ_1 pero altos en θ_2 tienen la misma probabilidad de acierto y viceversa.

Si θ_1 y θ_2 fuesen entendidas como habilidad visuoespacial y conocimientos en geometría, respectivamente, personas sin apenas habilidad visuoespacial tendrían la misma probabilidad de acierto que personas con un nivel de habilidad medio en las dos dimensiones.

Figura 2.5.1

Gráfico de contorno de la SCI



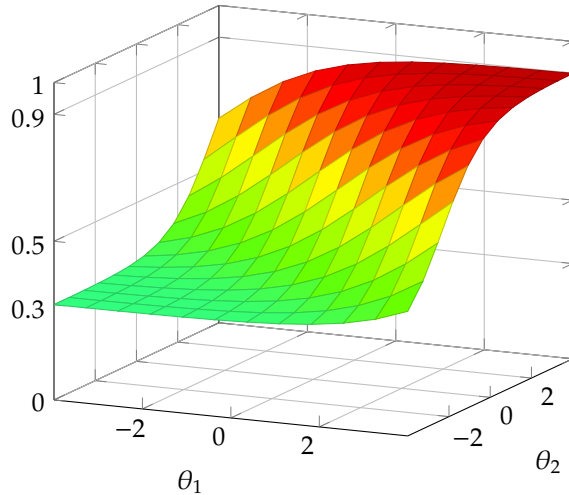
Nota. Las líneas de contorno muestran la probabilidad de acierto para diferentes niveles de θ .

$a_i \theta'_j + d_i$ define que hay un parámetro de discriminación a y un valor de θ para cada dimensión y a esto se le suma la constante d .

Estos modelos también pueden tener una adaptación al modelo 4PL⁹ o 3PL como se muestra en la siguiente figura.

Figura 2.6

SCI del modelo de cuatro parámetros



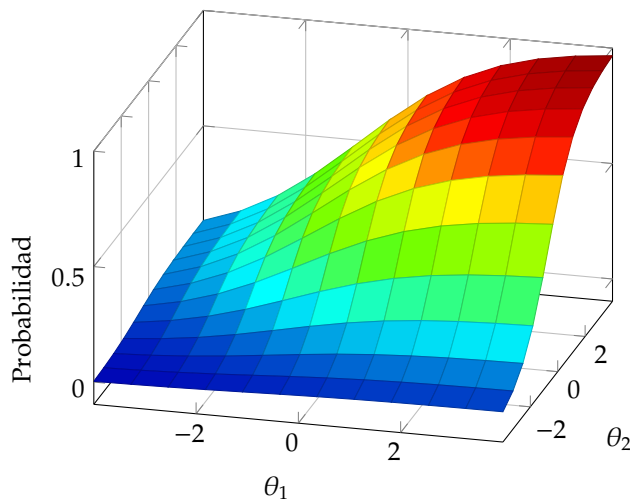
Nota. SCI de un ítem con $a_1 = .75$, $a_2 = 1$, $d = -.7$, $c = .3$ y $u = .9$.

Modelos parcialmente compensatorios basados en la TRI

A diferencia de los compensatorios, estos no entienden el conocimiento de una forma complementaria: los modelos parcialmente compensatorios establecen que para tener una alta probabilidad de acierto se deben tener altos niveles de conocimiento en cualquier dimensión.

Figura 2.7

SCI parcialmente compensatoria

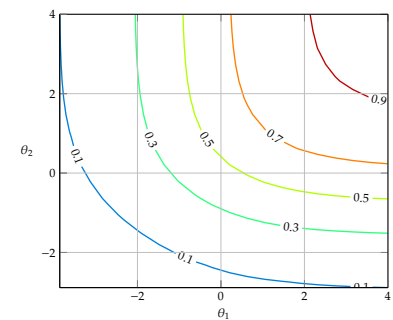


Nota. El gráfico muestra la SCI parcialmente compensatoria para los mismos parámetros que en la Figura 2.5.

9: En la TRI el cuarto parámetro estaba representado por d , pero en la MIRT para evitar confusiones se utilizará u .

Figura 2.7.1

Gráfico de contorno parcialmente compensatorio



Nota. Se puede apreciar como es necesario tener altos niveles de θ en cada dimensión para cada probabilidad de acierto.

La SCI, por tanto, cambia y pasa a definirse por

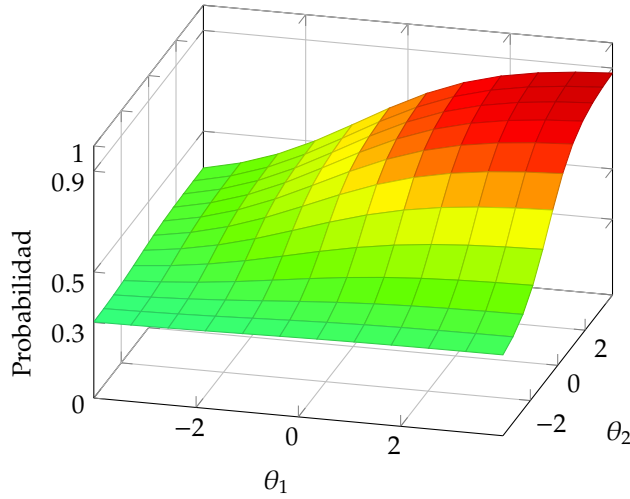
$$Pu_{ij}(\theta_j, a_i, d_i) = \left(\prod_{l=1}^m \frac{e^{a_{il}\theta'_{jl}+d_{il}}}{e^{a_{il}\theta'_{jl}+d_{il}}} \right) \quad (2.6)$$

Un modelo parcialmente compensatorio equivale a multiplicar la probabilidad de acierto de tantos modelos de TRI como dimensiones tenga el ítem y esto es lo que define $\prod_{l=1}^m$, donde l es cada dimensión y m el total de ellas. A diferencia de los modelos compensatorios, el nivel de conocimiento no se suma sino que se multiplica y, por lo tanto, ya no se está ante una relación lineal entre rasgos.

Estos modelos también tienen una extensión al modelo 2PL, 3PL y 4PL, como se aprecia en la siguiente figura.

Figura 2.8

SCI parcialmente compensatoria del modelo 4PL



Nota. El gráfico muestra la SCI parcialmente compensatoria para los mismos parámetros que el ítem de la Figura 2.6.

Obtención de parámetros

3

En este capítulo se van a mostrar todas las aplicaciones en R, de la teoría explicada en el capítulo anterior. Primero se mostrará cómo obtener los dos tipos de modelos de la MIRT que se pueden obtener con la librería *mirt* (Chalmers, 2012); estos no se clasifican, como en el capítulo anterior, según el número de parámetros que tiene la Curva Característica del Ítem (CCI), sino que se caracterizan por la estructura o el análisis factorial que permite cada función.

Al final del capítulo se evaluará, con diferentes pruebas estadísticas, cómo de bueno es el modelo que se ha obtenido. Esto también supondrá determinar cuál es el número de dimensiones que tienen los datos que se han visto en el Primer capítulo.

Introducción al Análisis Factorial

El Análisis Factorial (AF) es una técnica estadística multivariante que sirve para estudiar las dimensiones subyacentes entre varias variables (ítems), llamadas factores (Bruin, 2011). Su objetivo principal radica en encontrar el menor número de factores que sean capaces de explicar qué relación existe entre las variables de un conjunto de datos (Kline, 2014). Además de determinar el número de dimensiones, también se mide qué ítems son más característicos de cada factor y se pretende establecer una **estructura simple** (Lloret-Segura y col., 2014).

Existen dos tipos de Análisis Factorial: Análisis Factorial Exploratorio (AFE) y Análisis Factorial Confirmatorio (AFC), el primero se hace sin una asunción previa del número de factores y, por lo tanto, se usan técnicas que contemplan que cualquier ítem es propio de cualquier dimensión; el segundo, comprueba hipótesis previamente establecidas, basadas en cuál es el número de factores, si hay o no correlaciones entre estos y qué peso tiene cada ítem en cada factor (Abad y col., 2011), entre otras.

Un aspecto importante del AFE es el concepto de rotaciones, que se verá más adelante, basado en reposicionar factores de una forma más interpretable, que tengan sentido tanto a nivel matemático como a nivel teórico (Osborne, 2015). Gracias a esto se pueden definir escalas de Ansiedad, Esquizofrenia, etc, en un test psicológico, ya que se puede observar como ítems que fueron redactados para evaluar esos aspectos psicológicos, pesan más en un solo factor y no en otro, es decir, aquellos ítems forman parte de una dimensión abstracta que a posteriori se nombra según el criterio de personas expertas.

Dicho esto, los factores también pueden tener estructuras con diferentes niveles de profundidad entre factores (Mair, 2018), donde unos pueden ser más generales y simbolizar constructos como la habilidad matemática y otros subfactores como sería el álgebra, cálculo, etc.

Introducción al Análisis Factorial	14
3.1 Modelo exploratorio	15
3.2 Modelo confirmatorio	17
Modelos two-tier	18
3.3 Bondad de ajuste	19
Principio de monotonicidad	19
Análisis de Varianza (ANOVA)	20
Convergencia del modelo	21
itemfit	22
M2	22
residuals	23
clusters	24

Estructura Simple

En el AF se usa como criterio para determinar cómo de adecuada es una solución factorial. Requiere que cada ítem pese considerablemente en un solo factor y que estos muestren un patrón, repartidos por los demás factores.

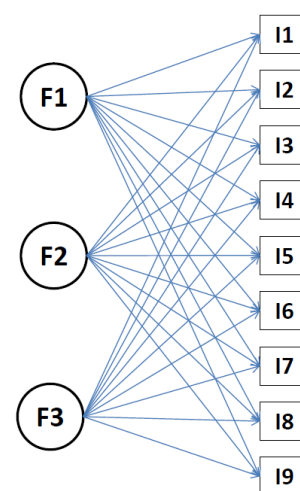
3.1. Modelo exploratorio

Estos modelos se estiman a través de la función *mirt()*, la cual tiene más de 30 argumentos mayormente relacionados con el método de optimización*; los más importantes se muestran a continuación:

- data** Una 'matrix' o 'data.frame' con valores numéricos y valores en blanco cifrados como 'NA'.
- model** Una 'string' que especifique un argumento de la función *mirt.model()* o un objeto de esta misma función, que especifique cómo debe ser estimado el modelo de la MIRT. Para modelos exploratorios solo es necesario especificar el número de dimensiones.
- itemtype** Una 'string' que determine el tipo de ítem que debe ser estimado:
 - '2PL', '3PL', '3PLu', '4PL' para los modelos compensatorios de 2 a 4 parámetros; 3PL solo estima la asíntota inferior y 3PLu la superior.
 - 'PC2PL', 'PC3PL' para los modelos de 2 a 3 parámetros parcialmente compensatorios. Este tipo de ítems solo pueden ser usados en modelos confirmatorios.
- guessing** Valor numérico para fijar el parámetro de *c* de pseudo-acierto para todos los ítems introduciendo o introducir un vector con el valor de cada ítem.
- upper** Valor numérico para fijar el parámetro de la asíntota superior en el modelo 4PL para todos o introducir un vector con el valor de cada ítem.
- method** Especificar el método de optimización, ya sea 'EM' (Máxima Verosimilitud), 'QMCEM' (casi-Monte Carlo EM), 'MCEM' (Monte Carlo EM), 'MHRM' (Metropolitan Hastings), 'SEM' (EM Estocástico) y 'BL' (Bock y Lieberman). 'EM' da buenos resultados cuando se estiman de 1 a 3 dimensiones y 'BL' no se recomienda para tests largos, los demás son más efectivos con más de 3 dimensiones y, por lo general, más rápidos que 'EM'.¹

Figura 3.1

Estructura no jerárquica



Nota. El diagrama muestra una solución factorial de tres factores, donde cada uno de los 9 ítems es representativo de cada factor. Además, ningún factor está correlacionado porque no se muestran flechas que los unan, por lo tanto, cada factor representa conocimientos o conceptos completamente diferentes.

1: Aunque haya una gran variedad, lo importante es que se llegue a una solución aceptable, es decir, que el algoritmo converja.

* Para más información mirara la documentación de la función («mirt: Full-Information Item Factor Analysis (Multidimensional Item Response Theory)», *s.f.*) y el paquete *mirt* (Chalmers, 2012), además de el Apéndice C para entender mejor el concepto de optimización.

El tiempo de estimación puede reducirse utilizando *mirtCluster()*, ya que permite aumentar el número de procesos que se dan simultáneamente. El siguiente código muestra como obtener un modelo 2PL, utilizando todos los núcleos del procesador excepto 1², a través del método 'MHRM', al final, se vuelve a restablecer el número de núcleos.

```
5 library(mirt)
6 library(parallel)
7
8 mirtCluster(detectCores()-1)
9
10 mirt.exp.2F <- mirt(df, 3, itemtype = '2PL', method = 'MHRM')
11
12 mirtCluster(remove = TRUE)
```

2: Con la función *detectCores()*, del paquete *parallel* (R Core Team, 2019), se obtiene cuántos núcleos tiene el ordenador en el que se trabaje. En este ejemplo se prescinde de 1 para que, mientras se obtiene el modelo, se puedan llevar a cabo otras tareas en el ordenador, sino, se bloquearía hasta que la función acabase.

Mientras *mirt* calcula los parámetros irá devolviendo a tiempo real valores que determinan cómo de bien se aproxima a una solución:

Stage 3 = 82, LL = -219879.5, AR(0.51) = [0.40], gam = 0.0066, Max-Change = 0.0007

El resultado de la función es el siguiente:

```
\lab{mirtresult}
  Call:
mirt(data = df, model = 3, itemtype = "2PL", method = "MHRM")

Full-information item factor analysis with 3 factor(s).
Converged within 0.001 tolerance after 82 MHRM iterations.
mirt version: 1.32.1
M-step optimizer: NR1
Latent density type: Gaussian
Average MH acceptance ratio(s): 0.396

Log-likelihood = -197099.2, SE = 0.072
Estimated parameters: 237
AIC = 394672.4; AICc = 394686.9
BIC = 396328.3; SABIC = 395575.2
G2 (1e+10) = 250403.2, p = 1
RMSEA = 0, CFI = NaN, TLI = NaN
```

Hecho esto, la función *coef()* extrae los parámetros de los ítems y es aquí donde la rotación de los factores se puede aplicar. Los factores, como se ha ido viendo hasta ahora, son conceptos abstractos que representan las relaciones entre ítems y a los que se les puede calcular el peso que tienen en cada factor; en la MIRT el peso de cada ítem equivale al parámetro de discriminación.

Las rotaciones, como se ha explicado anteriormente, buscan una estructura simple y, por lo tanto, deberían conseguir concentrar el peso de un ítem en un solo factor. Con esta premisa se han establecido muchos métodos durante la historia y se pueden categorizar según si las rotaciones son ortogonales o oblicuas y esto se traduce en si se asume que los factores pueden tener similitudes – oblicuos – o no – ortogonales – (Kline, 2014).

Por lo tanto, sería lógico pensar que las rotaciones oblicuas son mejores, ya que cuesta asumir que dos ámbitos de conocimiento de un mismo examen no tengan nada en común. Cabe destacar que hay una gran cantidad de rotaciones**, cuyas propiedades varían a nivel matemático y conceptual, dando la posibilidad de adaptarse mejor a los conocimientos o al planteamiento teórico que tenían las personas que redactaron esos ítems; por consiguiente, la elección de una rotación u otra debe ser acorde al criterio de las personas que elaboraron ese examen y de la interpretabilidad de esos parámetros rotados (Forina y col., 1989).

A continuación se muestran una rotación oblimin (oblicua) del modelo obtenido anteriormente, de la cual se extrae el primer ítem; por defecto `coef()` no hace ninguna rotación.

```
13 coef(mirt.exp.2F, rotate = 'oblimin')[[1]]

Rotation: oblimin

      a1      a2      a3      d g u
par 0.2579561 1.242415 0.6643525 5.830586 0 1
```

El output de la función `coef()` es una lista y este tipo de formato no es adecuado para otras funciones que se verán más adelante. Para pasar de lista a `data.frame`, se puede utilizar la función `CoefToDataframe()`:

```
14 parametros.mirt.exp.2F <- CoefToDataframe(n_items = 60, modelo =
  mirt.exp.2F, rotate = 'oblimin')
```

3.2. Modelo confirmatorio

En este método, como se ha explicado en la introducción al capítulo, se necesita especificar qué ítems son característicos de cada dimensión; para ello se utiliza la función `mirt.model()`, cuyos argumentos pueden determinar qué ítems van en cada dimensión, cuáles deberían tener el mismo parámetro de discriminación, cuáles están relacionados y que distribución de probabilidad tiene los parámetros de un modelo, entre otros.

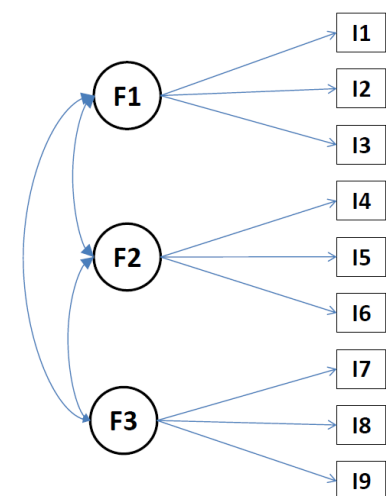
Aunque se podría introducir una opción por cada argumento, `mirt.model()` puede leer una 'string' donde se especifique toda la información necesaria. A continuación se muestra como especificar que cada dimensión contenga un tercio de los ítems y que estos no están presentes en ninguna otra, excepto el ítem 32; además, se establecen relaciones entre el segundo y el tercer factor.

```
15 modelo.TwoTier.3F <- mirt.model(
16     'F1 = 1-20
17     F2 = 21-40
18     F3 = 32, 41-60
19     COV = F2*F3'
20 )
```

** Para más información consultar el manual del paquete GPArotation (Bernaards & I.Jennrich, 2005), del cual se basa la función `coef()` para hacer rotaciones, y «Factor Analysis Rotation» (s.f.) para entender sus implicaciones conceptuales.

Figura 3.2

Modelo confirmatorio



Nota. El diagrama muestra una solución factorial de tres factores, donde cada uno de los 9 ítems es representativo de un solo factor. Además, todos los factores están correlacionados.

```
21 mirt.TwoTier.3F <- mirt(df, modelo, itemtype = '2PL', method = '
    MHRM')
```

Modelos two-tier

Este tipo de modelos especifican que las dimensiones subyacentes se dividen en uno o varios factores generales o primarios donde todos los ítems tienen un peso determinado y, además, estos ítems se dividen en otros factores específicos no relacionados entre ellos (Mair, 2018), como se observa en la Figura 2.3.

En cambio, el modelo bifactorial es un modelo two-tier específico que establece un solo factor general. Estos modelos se obtienen con la función *bfactor()*.

bfactor() tiene tres argumentos principales:

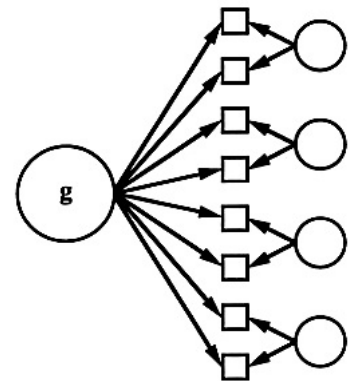
- data** Una 'matrix' o 'data.frame' con valores numéricos y valores en blancos cifrados como 'NA'.
- model** Un vector numérico que especifique qué factor contiene cada ítem. Por ejemplo, para un test de 4 ítems, donde el primer factor específico contiene los dos primeros y el segundo factor los restantes, el vector sería 'c(1, 1, 2, 2)'.
- model2** Un objeto definido por *mirt.model()* que especifique, para un modelo two-tier, qué ítems pesan en cada factor. Si se dejase en blanco, *bfactor* estima un modelo bifactorial por defecto.
- ...** Argumentos adicionales usados en la función *mirt* como 'itemtype', 'upper' o 'guessing', entre otros.

A continuación se muestra un ejemplo de un modelo two-tier con dos factores generales correlacionados, que contienen la primera y la segunda mitad de los ítems, respectivamente, y 4 específicos con el mismo número de ítems.

```
22 model.TwoTier.3F.tier <- mirt.model(
23     'F1 = 1-30
24     F2 = 31-60,
25     COV = F1*F2'
26 )
27
28 mirt.TwoTier.3F.tier <- bfactor(df, c(rep(1:4, each = 15)),
    modelo_two.tier, itemtype = '2PL')
```

Figura 3.3

Modelo bifactorial



Nota. El diagrama muestra una estructura bifactorial con un factor general *g*, 8 ítems representados por cuadrados y 4 factores específicos, no correlacionados, que contiene dos ítems cada uno. De *Intelligence-Two Models*. (1999, Enero). Intelligence. Consultado el 15 de abril de 2021, desde <http://www.personalityresearch.org/intelligence/structure.html>.

3.3. Bondad de ajuste

Hasta ahora solo se han comentado formas de obtener los parámetros de los ítems, pero no se ha ahondado en determinar cómo de bien se ajusta un modelo a la base de datos en la que se trabaje.

El resultado de `mirt.exp.2F`, antes visto, daba el valor de varios estadísticos de bondad de ajuste como RMSEA, AIC o BIC, entre otros; estos valores indican cómo de bueno es el modelo a nivel estadístico y, aunque estos en especial no tengan puntos de corte, hay otros métodos y funciones en el paquete `mirt` que los aprovechan para determinar su ajuste.

En esta sección se verán pruebas de bondad de ajuste y su interpretación para evaluar un modelo de la MIRT.

Bondad de ajuste

La bondad de ajuste es el grado en que los valores predichos por un modelo coinciden con los valores observados empíricamente. Un valor no significativo de una prueba estadística de bondad de ajuste indica que el modelo está bien ajustado.

Principio de monotonicidad

Aunque no sea una prueba de bondad de ajuste como tal, es necesario revisar qué valor tienen los parámetros de discriminación para cada dimensión. Como se ha explicado en el capítulo anterior, el principio de monotonicidad determina que a medida que aumenta el nivel de θ aumenta la probabilidad de acierto y esto se traduce en tener parámetros a_i positivos.

El modelo `mirt.exp.2F`, por ejemplo, no cumple este supuesto.

```
29 coef(mirt.exp.2F)
```

```
$I1
      a1      a2      a3      d g u
par -0.45 0.741 1.055 5.831 0 1

$I2
      a1      a2      a3      d g u
par -0.55 -0.615 0.71 3.808 0 1

$I3
      a1      a2      a3      d g u
par -0.258 -0.532 0.237 3.299 0 1

$I4
      a1      a2      a3      d g u
par -0.508 -0.405 -0.163 3.113 0 1

$I5
      a1      a2      a3      d g u
par -1.167 2.048 0.084 5.972 0 1

...      ...      ...

$I56
      a1      a2      a3      d g u
par -0.448 0.675 0.05 -0.405 0 1
```



```

$I57
      a1      a2      a3      d g u
par -0.206 0.173 -0.125 -0.721 0 1

$I58
      a1      a2      a3      d g u
par -0.009 -0.539 -0.201 -0.255 0 1

$I59
      a1      a2 a3      d g u
par -0.29 -0.354 0 -0.249 0 1

$I60
      a1 a2 a3      d g u
par -0.11 0 0 -0.907 0 1

$GroupPars
      MEAN_1 MEAN_2 MEAN_3 COV_11 COV_21 COV_31 COV_22 COV_32 COV_33
par      0      0      0      1      0      0      1      0      1

```

Aunque no se muestren todos, la mayoría tiene valores negativos. Esto indica que el modelo podría no estar bien especificado y se debería probar con otros modelos como los parcialmente compensatorios o no paramétricos^{***}, entre otros. Es extraño que, en prácticamente su totalidad, el test vulnere el principio de monotonicidad; en estos casos se deberían evaluar estos ítems y comprobar que no se hayan redactado de forma inversa (que empiezan con una negación) o que estén mal cifradas las respuestas en la base de datos (Kao, 2014).

A pesar de esto, se puede ignorar un valor de a_i negativo si se tiene una teoría o hipótesis muy fuerte que apoye el hecho de que personas con niveles más bajos de habilidad tengan mayor probabilidad de acertar el ítem.

Análisis de Varianza (ANOVA)

En la MIRT este método sirve para comparar los estadísticos de bondad de ajuste como RMSEA, AIC o BIC, de dos modelos previamente calculados, para establecer si el modelo más simple es más adecuado que otro más complejo. ANOVA, al ser una prueba estadística, tiene un valor p asociado que, si es inferior a 0.05, demuestra que el modelo más simple es mejor.

A continuación se observa como p es 0, por lo tanto, el modelo de 3 dimensiones `mirt.exp.2F` obtenido anteriormente se ajusta mejor que el de 4 dimensiones que se acaba de estimar.

```

30 mirtCompExp2PL.4 <- mirt(df, 4, itemtype = '2PL', method = 'MHRM')
31
32 anova(mirt.exp.2F, mirtCompExp2PL.4)

```

```

Model 1: mirt(data = df, model = 3, itemtype = "2PL", method = "MHRM")
Model 2: mirt(data = df, model = 4, itemtype = "2PL", method = "MHRM")

```

^{***} Para más información consultar «mirt: Full-Information Item Factor Analysis (Multidimensional Item Response Theory)» (s.f.).

	AIC	AICc	SABIC	HQ	BIC	logLik	X2	df	p
1	394685.2	394699.8	395588.1	395252.1	396341.2	-197105.6	NaN	NaN	NaN
2	394662.8	394685.3	395782.7	395365.9	396717.0	-197037.4	136.487	57	0

anova() es capaz de comparar cualquier modelo obtenido por la función *mirt()* o *bfactor()*, ya sea factorial o exploratorio, por lo tanto, las hipótesis que se tienen sobre como se distribuyen los ítems en cada factor pueden ser testadas. Por otro lado, la simplicidad de un modelo no viene dada solo por el número de dimensiones, la simplicidad puede ser entendida como el nivel de restricción. Ante la duda, *anova* ya especifica cuál es el modelo 1 y 2: si *p* es inferior a 0.05 se prefiere el modelo 1.

Convergencia del modelo

Otra forma de evaluar la bondad de ajuste es evaluando si se converge a una solución estable. Por ejemplo, el resultado del modelo two-tier mencionado anteriormente da el siguiente error:

```
Iteration: 84, Log-Lik: -199885.412, Max-Change: 0.54939
Warning messages:
1: Latent trait variance-covariance matrix became non-positive definite.
2: In log(eigen(sigma, symmetric = TRUE, only.values = TRUE)$values) :
   NaNs produced
3: In log(eigen(sigma, symmetric = TRUE, only.values = TRUE)$values) :
   NaNs produced
4: In log(eigen(sigma, symmetric = TRUE, only.values = TRUE)$values) :
   NaNs produced
5: In log(eigen(sigma, symmetric = TRUE, only.values = TRUE)$values) :
   NaNs produced
6: In log(eigen(sigma, symmetric = TRUE, only.values = TRUE)$values) :
   NaNs produced
7: Log-likelihood was decreasing near the ML solution. EM method may be unstable
```

anova también muestra que hacer comparaciones con modelos mal especificados no es posible:

```
33 anova(mirt.exp.2F, mirt.TwoTier.3F)
```

```
Model 1: bfactor(data = df, model = c(rep(1:4, each = 15)), model2 = modelo_two.tier,
  itemtype = "2PL")
Model 2: mirt(data = df, model = 3, itemtype = "2PL", method = "MHRM")
```

	AIC	AICc	SABIC	HQ	BIC	logLik	X2	df	p
1	NA	NA	NA	NA	NA	NA	NaN	NaN	NaN
2	394685.2	394699.8	395588.1	395252.1	396341.2	-197105.6	NA	56	NA

itemfit

Esta función puede estimar una gran variedad de pruebas estadísticas para determinar el ajuste de cada ítem al modelo y así determinar cuáles son problemáticos. Aquí se propone la prueba 'S_X2', que tiene por defecto itemfit, ya que Kang y Chen (2008) demostraron una mayor sensibilidad de 'S_X2' sobre 'Q2' o 'G2' – dos de los más utilizados hasta el momento – que también se pueden implementar con itemfit.

```
34 itemfit(mirt.exp.3F)
```

	item	S_X2	df.S_X2	RMSEA.S_X2	p.S_X2
1	I1	18.130	14	0.006	0.201
2	I2	25.205	20	0.006	0.194
3	I3	19.977	19	0.003	0.396
4	I4	21.279	21	0.001	0.442
5	I5	24.907	18	0.007	0.127
		
27	I27	35.510	19	0.010	0.012
		
55	I55	16.211	23	0.000	0.846
56	I56	25.894	22	0.005	0.256
57	I57	31.515	22	0.007	0.086
58	I58	23.753	23	0.002	0.418
59	I59	13.389	22	0.000	0.922
60	I60	23.727	23	0.002	0.419

Valores de 'p.S_X2' inferiores a 0.05 indican un ítem mal ajustado al modelo (Kang & Chen, 2008), por ejemplo, el ítem 27 muestra un mal ajuste, por lo tanto, se debería valorar su causa, como una mala redacción del ítem.

M2

M2 es una prueba estadística de la familia de las pruebas M_r , propuesta por Maydeu-Olivares y Joe (2005) para modelos con respuesta binaria. A diferencia de itemfit, este es un estadístico que evalúa el ajuste del test a los datos y, para ello, no los utiliza: obtiene parámetros a través de divisiones de los datos, cosa que permite hacer estimaciones de bases de datos pequeñas a costa de tiempo y perder información sobre la verdadera adecuación del modelo (Xu y col., 2017).

Su función debe contener un modelo mirt ya calculado y, si tuviese varias dimensiones, se debe especificar que se use una integración casi-Monte Carlo con el argumento **QMC**.

```
35 M2(mirt.exp.3F, QMC = TRUE)
```

	M2	df	p	RMSEA	RMSEA_5	RMSEA_95	SRMSR	TLI	CFI
stats	2567.397	1593	0	0.008744645	0.00811825	0.009361312	0.01329049	0.964936	0.9684424

M2 rechaza con mucha facilidad modelos bien ajustados como en este caso, ya que el valor de p es inferior a 0.05, por ello, se adjuntan otros estimadores de bondad de ajuste como RMSEA, TLI o CFI. Para modelos con ítems de respuesta binaria, Maydeu-Olivares y Joe (2014) propusieron que un RMSEA inferior a .05 o .089 indica un ajuste bueno o aceptable, respectivamente, pero Xu y col. (2017) observaron que el punto de corte debe establecerse en 0.03, por otro lado, valores de TLI o CFI por encima de 0.9 y 0.95 indican un ajuste aceptable o relativamente bueno, respectivamente.

Aunque no haya puntos de corte bien establecidos el modelo presenta un buen ajuste, incluso para aquellos más conservadores.

residuals

La función residuals permite evaluar el principio de independencia local visto en la sección de Principios. Para ello se evalúa el grado de relación entre pares de ítems, obteniendo de esta forma aquellos que no cumplen el principio de independencia local.

La función necesita un modelo mirt, especificar que se quieren valores p para cada par de ítems con el argumento `df.p` y, si tuviesen varias dimensiones, especificar que se use una integración casi-Monte Carlo con el argumento `QMC`.

```
36 residuales <- residuals(mirt.exp.3F, df.p = TRUE, QMC = T)
```

El output de la función son dos matrices, la primera tiene su triángulo superior con el valor de p de cada par de ítems y la segunda un valor de la V de Cramér estandarizado de -1 a 1 que indica con qué fuerza se vulnera la independencia local: valores cercanos a los extremos indican una clara vulneración y los cercanos a 0 una vulneración prácticamente nula (Allen, 2017).

Para saber qué ítems tienen valores de p inferiores a 0.05 y observar sus V de Cramér se puede emplear la siguiente solución, aunque hay muchas otras formas de indexar valores en R.

```
37 cbind(which(residuales$df.p < 0.05, arr.ind = T), VCramér =
      resid$LD[which(residuales$df.p < 0.05)])
```

	row	col	VCramér
I6	6	7	0.02347876
I5	5	12	0.02723153
I8	8	16	0.02570147
I11	11	16	0.02833874
I14	14	17	0.02393573
	
I43	43	60	0.02360453
I47	47	60	0.02538431
I48	48	60	-0.03814862
I58	58	60	-0.02653811
I59	59	60	0.04918944

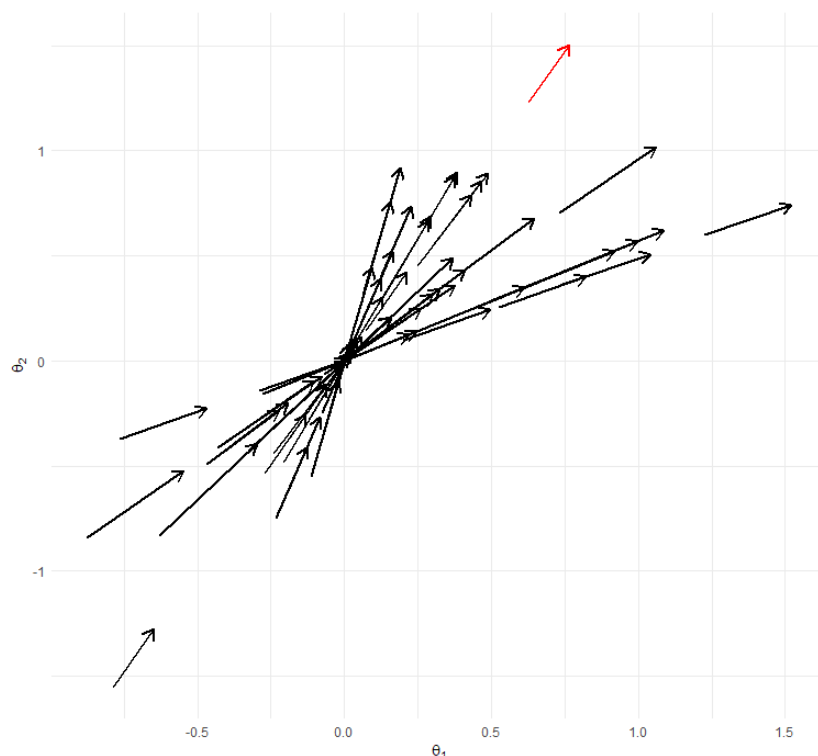
En total 59 parejas de ítems vulneran el principio de independencia local según el valor de p pero la V de Cramér estandarizada muestra una evidencia prácticamente nula, cosa que demuestra cómo de sensible puede llegar a ser p .

clusters

Reckase (2009) propuso que el máximo de factores de un modelo no puede exceder el número de clusters que tienen los datos. Esos clusters se obtienen, primero, definiendo el ángulo que forman los ítems cuando se representan en un sistema de coordenadas polares³ como se muestra a continuación.

Figura 3.4

Sistema de coordenadas polares



Nota. Cada vector corresponde a un ítem, cuya dirección equivale a la dirección de máxima pendiente del ítem, la distancia entre su origen y el punto [0 0] equivale a MDIFF y la longitud del vector equivale a MDISC. El vector rojo corresponde a un ítem con $a1 = 1.0293996$, $a2 = 1.8069402$, $d = -2.54934277$, $MDIFF = 1.22588692$ y $MDISC = 2.0795905$. Se puede observar como el vector es más paralelo a θ_2 , ya que es más discriminativo para esta dimensión.

3: Es un sistema de coordenadas que define cada punto en un plano o sistema multidimensional por una distancia y un ángulo respecto a un eje.

Mirar el Apéndice B para entender mejor los conceptos MDIFF y MDISC.

Por otro lado, se utiliza el método de Ward para estimar clusters a partir de esos ángulos. La función `ward_mat()` obtiene los ángulos entre cada par de ítems e implementar el algoritmo que obtiene los clusters se consigue con `hclust()`, pero antes se deben obtener los ángulos de cada ítem respecto a cada eje con la función `polar_coords()`.

```
38 coordenadas.mirt.exp.3F <- polar_coords(parametros.mirt.exp.3F
    [,1:3], parametros.df$d)
```

```
39
```

```

40 ward.mirt.exp.3F <- ward_mat(coordenadas[c('ang1', 'ang2', 'ang3')
41                               ])
42 clusters.mirt.exp.3F <- hclust(as.dist(ward.mirt.exp.3F), method =
43                               'ward.D2')

```

Si se visualiza un dendrograma a través de *plot* y se especifica que se muestren tres clusters con *rect.hclust()* se puede observar como hay 3 clusters principales.

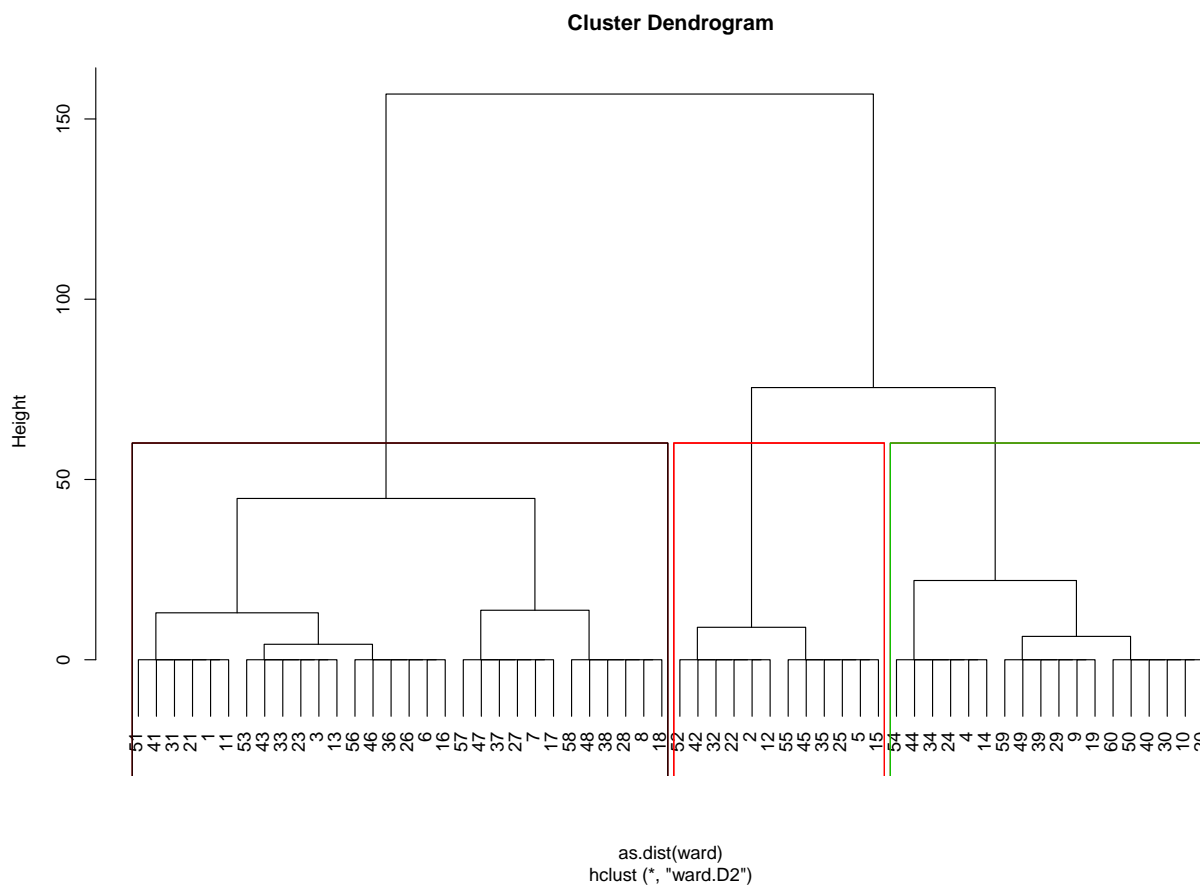
```

43 plot(clusters, cex = 0.6)
44
45 rect.hclust(clusters, k = 3, border = 1:3)

```

Figura 3.5

Dendrograma



Nota. La altura (Height) indica el orden en que se han hecho los clusters. Primero se obtienen dos clusters con la misma distancia que se dividen en otros dos clusters cada uno.

El punto de corte que toma Reckase (2009) estaría inmediatamente debajo de la primera división de la derecha, dejando por debajo a los dos clusters de la izquierda, de esta forma, habría un máximo de tres líneas verticales que equivalen al número de clusters.

La principal premisa de un TAI es seleccionar los ítems que se presentan a una persona de forma dinámica a medida que avanza en el examen y ese dinamismo supone seleccionar el próximo ítem acorde a las respuestas que ha dado la persona en los anteriores y considerando la información que estos dan respecto al nivel de habilidad de la persona (Oppl y col., 2017). Además, los ítems que se administran en un TAI deben provenir de una banco de ítems calibrado (Vale, 2006/2015), que es en sí mismo, lo que se ha procurado obtener durante los capítulos anteriores.

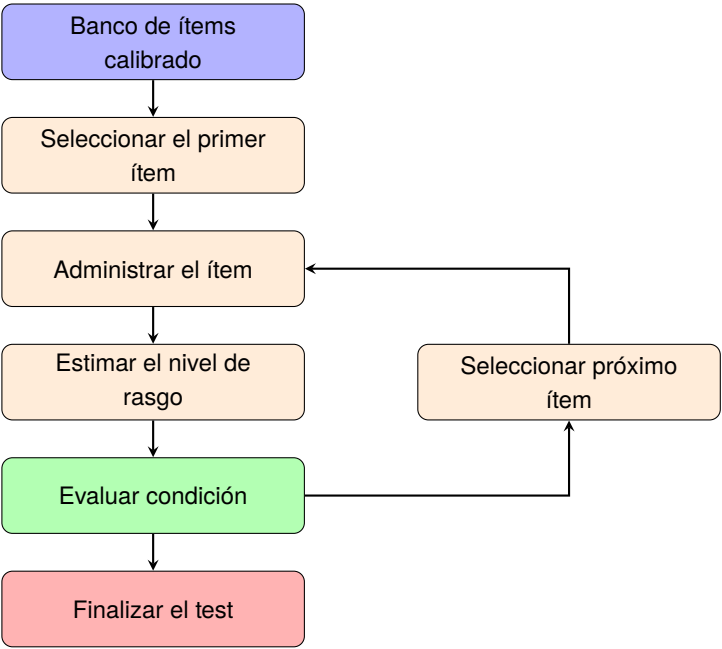
La administración empieza por seleccionar un ítem al azar o uno que tenga una dificultad media, ya que en este estadio, no se tiene ninguna información sobre la administración de ítems previos. Después, en el momento en que se tiene la suficiente información sobre la persona para estimar el nivel de rasgo, la selección del próximo ítem se basa en métodos estadísticos más sofisticados, como el del Apéndice D.

Cada vez que se da una respuesta se actualiza el nivel de rasgo y el TAI finaliza cuando se cumple una condición o varias, como pueden ser la largada del test o el error de medida del nivel de rasgo de una persona. Cabe destacar que un TAI no acepta respuestas en blanco y, por lo tanto, la persona que responda se ve obligada a seleccionar una opción; en estos casos los ítems no respondidos se suelen considerar fallados (Mills & Stocking, 1996).

A continuación se muestra un diagrama de todos los pasos:

Figura 4.1

Pasos de un TAI



4.1 mirtCAT	27
Resultados	30
4.2 Consideraciones	33
4.3 Ejemplo	33

4.1. mirtCAT

En R, el paquete mirtCAT (Chalmers, 2016) y su función, también llamada *mirtCAT()*, consiguen implementar un TAI a través de una interfaz (GUI) en HTML con shiny y también ubica todos los procesos y cálculos en el ordenador local donde se llame a la función mirtCAT. Esta función tiene una gran cantidad de argumentos y opciones, repartidos entre diseño del TAI y de la GUI, selección del ítem e incluso se puede administrar un preTAI¹.

Su funcionalidad se verá a partir de ahora, pero se recomienda consultar Chalmers (2016) para ampliar la información que aquí se proporciona. Los argumentos de mirtCAT se irán presentando a continuación con explicaciones y ejemplos basados en Chalmers (2016).

df Un data.frame que contenga los enunciados (*Question*), opciones a los ítems (*Option*), respuestas (*Answer*), tipos de opciones (*Type*) y enlaces a material adicional que se quiera introducir en el enunciado como imágenes o tablas (*Stem*).

A continuación se muestra un ejemplo de un test no adaptativo de tres ítems con dos opciones de respuesta, donde los dos primeros tienen casillas para cada opción de respuesta y el último un botón, la primera opción es la correcta y el último enunciado contiene una imagen.

```
46 enunciados <- c('¿2 + 3 = 5?', '¿Preceder significa estar o ir
    antes en el tiempo o delante en el espacio de otra persona o
    cosa que se toma como referencia, en especial si existe una
    anterioridad inmediata?', '¿La imagen de arriba muestra un
    caribú?')
47
48 opciones <- matrix(c('Verdadero', 'Falso'), nrow = 3, ncol = 2,
    byrow = T)
49
50 respuestas <- rep('Verdadero', 3)
51
52 tipo <- c(rep('checkbox', 2), 'radio')
53
54 imagenes <- c('', '', 'D:/Documentos/TFG/Stems/Caribú.html')
55
56 df <- data.frame(Questions = enunciados,
57                     Option = opciones,
58                     Answer = respuestas,
59                     Type = tipo,
60                     Stem = imagenes)
61
62 TAI <- mirtCAT(df = df)
```

Siempre que se quiera adjuntar una imagen, una tabla o un enunciado hecho en un editor de textos, este debe estar en formato HTML (.html) o markdown (.md) y se debe especificar dónde está el archivo en el ordenador.

Automáticamente, la interfaz se abre en un buscador como Chrome o Firefox; es recomendable guardar el TAI en un objeto, en este ejemplo 'TAI', para poder obtener el nivel de rasgo o su error de medida a cada pregunta, como se verá más adelante. La sesión acaba cuando se cierra la

1: Pequeño número de ítems que se administra antes del TAI para tener un nivel de rasgo inicial y evitar problemas de cálculo con los algoritmos.

ventana en el buscador y automáticamente se guarda toda la información en el objeto asignado.

Figura 4.2

Interfaz de mirtCAT

mirtCAT

Authors:
Author information

To progress through the interface, click on the action button below.

Next



¿La imagen de arriba muestra un caribú?

- ☐ Verdadero
☐ Falso

Nota. Se puede observar como las opciones de respuesta del ítem tienen forma de botón y la imagen queda arriba del enunciado.

Dicho esto, los argumentos para hacer una evaluación adaptativa, a parte de **df** son:

- mo** Un objeto definido por la función `mirt` o `bfactor`.
- method** Método por el cual se estima el nivel de habilidad.
- criterio** Criterio por el cual se selecciona el proximo ítem.
- preCAT** Especificar el diseño de TAI previo a la administración de los ítems del test (preTAI).
- design** Especificar el criterio de finalización del TAI, entre otros.

El método por el cual se estima el nivel de habilidad puede cambiar el valor de θ final, haciéndolo más cercano o más alejados a valores cercanos a 0. Los métodos por máxima verosimilitud tienden a dar valores más extremos, como son 'ML' o 'WLE', aunque entre ellos también haya diferencias, 'WLE' da valores más conservadores que 'ML' aunque den el mismo error de medida («Warm's Mean Weighted Likelihood Estimates (WLE) of Rasch Measures», [s.f.](#)). Además, son incapaces de converger cuando se tiene un patrón de respuestas extremo, como alguien que lo contesta todo correctamente o incorrectamente.

Ante esto, surgieron los modelos Bayesianos, que siempre convergen a una solución aunque haya respuestas extremas, pero también produce que los valores de θ tiendan a valores más centrales (Abad y col., [2011](#)).

Los métodos Bayesianos son 'MAP' o 'EAP' y estos también difieren al igual que 'ML' y 'WLE': MAP da valores más centrales que 'EAP'. Los más comunes son 'MAP' y 'ML', el segundo no tendría que fallar si el test es lo suficientemente largo, pero sino convergiera, `mirtCAT()` lo cambia automáticamente por 'MAP'.

Por otro lado, los criterios de selección del próximo ítem para TAIs multidimensionales están basados en la estimación de divergencia de Kullback-Leibler: 'KL', 'KLD', y en la matriz de información de Fischer: 'Drule', 'Trule', 'Arule', 'Erule' y 'Wrule', entre otros.

Es lógico pensar que uno de ellos debe prevalecer sobre los demás a la hora de obtener estimaciones precisas, pero todavía es necesario hacer más investigación. Wang y Chang (2011) demostraron que, en modelos de dos dimensiones, el mejor criterio es el Método de Información Mútua (MUI), el cual no contempla *mirtCAT()*, pero sí da resultados similares a 'Drule'; además, se demostró que 'KL' es problemático cuando los valores de θ de una persona están cercanos a 0.

Esto indica que en modelos de dos dimensiones son preferibles los criterios relacionados con la matriz de información, pero que todavía está por descubrir el desempeño de los demás criterios en una gran variedad de modelos.

preCAT tiene muchos inputs que deben ser especificados en forma de lista:

- min_items** Mínimo número de ítems que presentar antes de empezar el TAI.
- max_items** Máximo número de ítems que presentar antes de empezar el TAI.
- criteria** Criterio de selección del ítem. Por defecto es aleatorio.
- method** Método para estimar el nivel de habilidad.
- response_variance** Detener preCAT si se estuviese llegando a una estimación precisa del nivel de habilidad. Por defecto no se detiene y necesita un valor lógico.

design, por otro lado, también tiene múltiples inputs, pero los más importantes para determinar una regla de parada son:

- min_SEM** Error de medida estándar para cada dimensión, por defecto es 0.3. Este valor define que el test se detenga cuando se tenga la certeza de que el nivel de habilidad no varía más de 0.3 o -0.3 unidades.
- delta_thetas** Un valor x que determina si el cambio en el nivel de habilidad, a medida que se presentan ítems, es significativo.
- min_items** Mínimo número de ítems que debe tener el TAI.
- max_items** Máximo número de ítems que debe tener el TAI.
- content** Un vector de caracteres que defina qué concepto evalúa cada ítem.
- content_prop** Definir la proporción de ítems que se quieren mostrar por cada concepto que evalúa el ítem, definido en el input anterior.
- classify** Un valor numérico como punto de corte para definir personas que están por encima o por debajo de un valor de θ determinado.

classify_CI Un valor numérico entre 0 y 1 que indique el intervalo de confianza para clasificar según el punto de corte especificado en `classify`.

exposure Valores entre 0 y 1 para el método Synpson-Hetter.

Unos de los mayores problemas de un TAI es administrar siempre los mismos ítems de un banco de ítems y estos siempre son los más informativos ya que los criterios de selección buscan ese tipo de ítems.

`mirtCAT()` implementa el de Synpson-Hetter basado en determinar el porcentaje de personas a las que se les debe presentar un ítem: 0.02 significaría que solo un 2 % de todas las personas examinadas podrían responder a ese ítem (Thompson, 2018).²

El último argumento que se presenta aquí es `shinyGUI`, cuyos inputs principales definen el estilo de la interfaz:

title Un vector de caracteres para el título de la GUI.

authors Un vector de caracteres para los nombres de de los autores o autoras o la institución del TAI. Puede omitirse para ignorar este input.

instructions Un vector de caracteres de tres componentes que indique como progresar por la GUI.

begin_message Texto que mostrar en la página anterior al inicio del TAI.

max_time Máximo de tiempo, en segundos, para finalizar el test. Por defecto no hay límite.

2: Cada vez que se busca un nuevo ítem en el banco, se genera un valor aleatorio entre 0 y 1, si ese valor es mayor que **exposure** no se presentará, si es inferior sí.

Resultados

Una vez finaliza el TAI, toda la información se guarda en el objeto al que se haya asignado. Para obtener el nivel de la persona y su error de medida solo hay que ejecutar ese objeto, pero para observar qué ítems se han administrado o que error de medida se tenía en cada administración se debe utilizar la función `summary()` y, adicionalmente, se puede visualizar con `plot()`.

63 `summary(TAI)`

```
$final_estimates
      Theta_1 Theta_2 Theta_3
Estimates 4.935148 19.99999 -2.8926607
SEs       0.000000  0.00000  0.1887483

$raw_responses
[1] "Verdadero" "Falso"     "Verdadero" "Falso"     "Falso"     "Falso"     "Verdadero"
[8] "Verdadero" "Verdadero" "Falso"     "Verdadero" "Verdadero" "Falso"     "Verdadero"
[15] "Falso"     "Verdadero" "Verdadero" "Falso"     "Verdadero" "Verdadero" "Falso"
[22] "Verdadero" "Verdadero" "Verdadero" "Falso"     "Verdadero" "Verdadero" "Verdadero"
[29] "Verdadero" "Verdadero" "Falso"     "Verdadero" "Verdadero" "Falso"     "Verdadero"
[36] "Verdadero" "Verdadero" "Verdadero" "Falso"     "Verdadero" "Verdadero" "Verdadero"
[43] "Falso"     "Verdadero" "Falso"     "Verdadero" "Falso"     "Verdadero" "Falso"
```

```
[50] "Verdadero" "Falso"
```

```
$scored_responses
```

```
[1] 1 1 1 0 0 1 0 0 1 1 1 1 0 1 0 1 1 0 0 0 0 0 0 0 1 1 0 1 1 1 0 0 0 1 1 1 0 0 0 1 1 1 1
[44] 1 1 0 0 1 0 1 0
```

```
$items_answered
```

```
[1] 1 32 29 54 39 13 30 57 31 4 19 37 46 2 23 41 55 3 27 43 53 40 15 22 34 9 28 60 33
[30] 14 16 42 59 11 18 38 51 6 20 45 50 36 7 24 35 8 25 47 44 10 26
```

```
$thetas_history
```

	Theta_1	Theta_2	Theta_3
[1,]	0.0000000000	0.0000000000	0.0000000000
[2,]	0.0007502515	0.003615403	0.001933021
[3,]	-0.0800203060	0.042610463	-0.065282892
[4,]	-0.1054463927	0.065133342	0.009937181
[5,]	-0.1054463927	0.065133342	0.009937181
...
[48,]	4.9351476198	19.999993297	-2.892660690
[49,]	4.9351476198	19.999993297	-2.892660690
[50,]	4.9351476198	19.999993297	-2.892660690
[51,]	4.9351476198	19.999993297	-2.892660690
[52,]	4.9351476198	19.999993297	-2.892660690

```
$thetas_SE_history
```

	Theta_1	Theta_2	Theta_3
	1.0000000	1.0000000	1.0000000
	0.9999040	0.9977713	0.9993632
	0.9806851	0.9932778	0.9860766
	0.9768578	0.9905754	0.9539358
theta_SE	0.9768578	0.9905754	0.9539358
	21.2878262	0.0000000	9.9201113
	2.2782406	0.0000000	4.4953453
	1.5733353	0.0000000	4.4395816
	1.5733293	0.0000000	4.3696122
...
	1.2117508	0.0000000	1.5042163
	1.2117455	0.0000000	1.5042015
	1.2113548	0.0000000	1.4926694
	1.2113547	0.0000000	1.4926694
	0.0000000	0.0000000	0.1887483

```
$terminated_sucessfully
```

```
[1] TRUE
```

```
$item_time
```

[1]	3.22	1.44	2.17	9.76	10.03	4.50	4.79	8.87	2.17	1.89	1.98	2.01	2.67	1.49
[15]	1.48	1.81	3.58	2.80	1.56	1.53	1.75	2.36	1.94	1.95	1.67	1.56	1.28	1.68
[29]	1.45	1.54	1.74	1.56	1.61	1.50	2.52	1.61	1.47	1.47	1.38	1.45	1.92	2.36
[43]	1.56	2.13	1.72	1.41	1.32	1.85	2.11	2.47	1.87					

Se puede observar como, tanto el error de medida como los niveles de θ final son inverosímiles y esto es debido a que se ha contestado al azar, además, los valores de θ negativos van en consonancia con los valores de discriminación negativos vistos en el Capítulo 3.

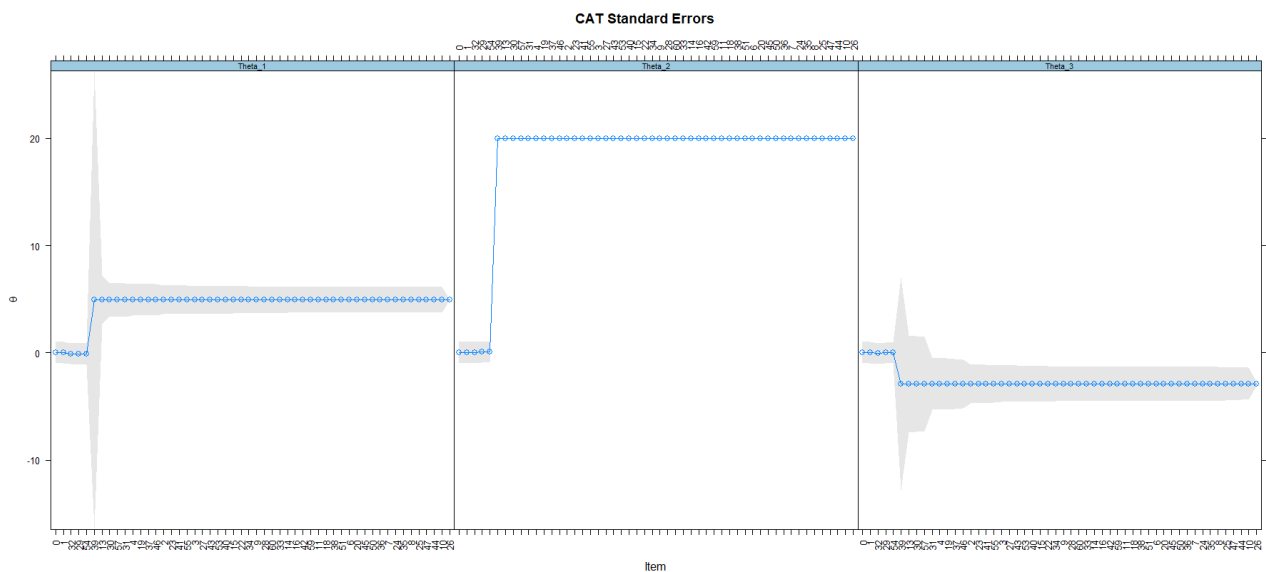
Por lo que respecta al preTAI, solo el apartado '\$thetas_SE_history' muestra el error de medida de los 3 ítems que se han administrado antes de empezar el TAI.

Con la función `plot()` es más fácil observar si hay algún patrón de respuestas extraño como se muestra a continuación.

64 `plot(TAI)`

Figura 4.3

Nivel de habilidad



Nota. La imagen muestra como cambia el nivel de habilidad de la persona, junto a su error de medida (en gris), para todas las dimensiones que evalúa TAI, a medida que la persona avanza por este último.

4.2. Consideraciones

En el inicio del Capítulo 3 se explicó qué es un Análisis Factorial Exploratorio (AFE) y Confirmatorio (AFC) como dos procesos aparentemente separados: si no conoces la distribución de los ítems en cada dimensión o el número de dimensiones conduces un AFE y si tienes una hipótesis clara sobre estos principios conduces un AFC.

En los últimos años estos análisis pueden ser entendidos como un proceso, donde primero se lleva un AFE con una mitad de los datos y después se prueban las hipótesis resultantes del AFE con un AFC en la otra mitad como se ve en Cabrera-Nguyen (2010) y Schreiber y col. (2006). En otras publicaciones se puede entender que no hay una norma clara a la hora de utilizar un AFE o un AFC de una forma determinada (Velada y col., 2009); que es mejor un AFE si un AFC no da un buen ajuste (Schmitt, 2011); que un AFE puede ser suficiente en muchos casos (Worthington & Whittaker, 2006) o que siempre se debe hacer un AFC para validar una escala o un modelo (Cabrera-Nguyen, 2010), entre otros.

mirtCAT() asume que cualquier modelo que se utilice para un TAI debe ser confirmatorio como se puede apreciar a continuación, con el modelo *mirt.exp.3F* obtenido en el Capítulo 3:

```
65 mirtCAT(df = df, mo = mirt.exp.3F, method = 'MAP', criteria = '
    Drule',
66          design = list(max_items = 10))
```

Error: CATs are intended for confirmatory IRT models not exploratory

Si se quisiera seguir con un modelo exploratorio e incluso unos parámetros que cumplan una determinada rotación, ya sea de un modelo confirmatorio o exploratorio, se puede utilizar la función *generate.mirt_object*, que tiene como primer argumento un *data.frame* donde se especifica qué valor tiene cada parámetro y qué tipo de ítem se quiere obtener; aunque primero hay que convertir el output de la función *coef* a *data.frame* como en el Capítulo 3.

```
67 parametros.mirt.exp.2F <- CoefToDataframe(n_items = 60, modelo =
    mirt.exp.2F, rotate = 'oblimin')
68
69 mirt2PL.coef <- generate.mirt_object(parameters = parametros.mirt.
    exp.2F,
70                                     itemtype = '2PL')
```

4.3. Ejemplo

A continuación se muestra un TAI con el modelo *mirt2PL* con una rotación varimax (ortogonal) con enunciados indeterminados para cada uno de los 60 ítems, repartidos en 4 contenidos diferentes: Comprensión lectora y oral, Ortografía y Cohesión, con la siguiente proporción de aparición: 25 %, 25 %, 30 % y 20 %, respectivamente.

Las opciones de respuesta son dicotómicas, ninguna necesita un archivo externo para el enunciado y las respuestas correctas son aleatorias. Se

utiliza un criterio 'Drule' para el próximo ítem y una estimación 'ML', al igual que en el preTAI, que tiene un máximo de 5 ítems.

El criterio de finalización es un error de medida mínimo de 0.2 y una largada máxima de 50 ítems, si se da una de estas dos premisas el TAI finalizará, además, se controla la exposición del ítem en un 80 %.

También se proporcionarán un título, unas instrucciones adecuadas, el nombre del autor y un máximo de una hora.

```

71 # Generar un mirt_object con mirt2PL
72
73 n_items <- 60
74
75 parametros.lst <- coef(mirt2PL, rotate = 'varimax')
76
77 parametros.df <- rbind(parametros.lst[[1]])
78
79 for(i in 2:n_items){
80
81   parametros.df <- rbind(parametros.df, parametros.lst[[i]])
82
83 }
84
85 mirt2PL.coef <- generate.mirt_object(parameters = parametros.df,
86                                     itemtype = '2PL')
87
88 # Definir df
89
90 enunciados <- sprintf('Enunciado.%d', seq_len(n_items))
91
92 opciones <- matrix(c('Verdadero', 'Falso'), nrow = 60, ncol = 2,
93                   byrow = T)
94
95 respuestas <- sample(c('Verdadero', 'Falso'), 60, replace = T)
96
97 tipo <- c('radio')
98
99 df <- data.frame(Questions = enunciados,
100                Option = opciones,
101                Answer = respuestas,
102                Type = tipo,
103                stringsAsFactors = F)
104
105 # Definir preCAT
106
107 preTAI <- list(max_items = 5,
108               criteria = 'Drule',
109               method = 'ML')
110
111 # Definir design
112
113 contenido <- rep(c('ComprensiónLectora', 'ComprensiónOral',
114                   'Ortografía', 'Cohesión'), each = 15)
115
116 contenido_prop <- c(ComprensiónLectora = 0.25,
117                   ComprensiónOral = 0.25,
118                   Ortografía = 0.30,
119                   Cohesión = 0.20)

```

```
119
120 diseño <- list(min_SEM = 0.2,
121               max_items = 50,
122               content = contenido,
123               content_prop = contenido_prop,
124               max_time = 3600,
125               exposure = 0.80)
126
127 # Definir ShinyGUI
128
129 GUI <- list(title = 'TAI: Ejemplo 1',
130           authors = 'Carlos Pérez',
131           instructions = c('Para contestar a un ítem se debe
132                           seleccionar siempre una de las dos opciones,
133                           no se permiten respuestas en blanco y
134                           finalizará en una hora.', 'Siguiendo ítem'))
135 # Definir mirtCAT
136
137 TAI <- mirtCAT(df = df, mirt2PL.coef, method = 'ML', criteria = '
138               Drule',
139               preCAT = preTAI, design = diseño, shinyGUI = GUI)
```


Bibliografía

- Abad, F. J., Olea, J., Ponsoda, V. & García, C. (2011). Síntesis. <https://www.sintesis.com/metodolog%C3%ADa%20de%20las%20ciencias%20del%20comportamiento%20y%20de%20la%20salud-22/medici%C3%B3n%20en%20ciencias%20sociales%20y%20de%20la%20salud-ebook-1572.html>
- Algoritmo esperanza-maximización. (2020). Wikimedia Foundation. https://es.wikipedia.org/wiki/Algoritmo_esperanza-maximizaci%C3%B3n
- Allen, M. (2017). Phi Coefficient, 4. <https://doi.org/10.4135/9781483381411>.
- Allison, P. (2014, 25 de septiembre). *Sensitivity Analysis for Not Missing at Random*. <https://statisticalhorizons.com/sensitivity-analysis>
- American Psychological Association. (s.f.). *APA Dictionary of Psychology*. Consultado el 24 de marzo de 2021, desde <https://dictionary.apa.org/clustered-data>
- Barton, M. A. & Lord, F. M. (1981). AN UPPER ASYMPTOTE FOR THE THREE-PARAMETER LOGISTIC ITEM-RESPONSE MODEL*. *ETS Research Report Series*, 1981(1), i-8. <https://doi.org/10.1002/j.2333-8504.1981.tb01255.x>
- Bernaards, C. A. & Jennrich, R. (2005). Gradient Projection Algorithms and Software for Arbitrary Rotation Criteria in Factor Analysis. *Educational and Psychological Measurement*, 65, 676-696.
- Bolt, D. M. & Lall, V. F. (2003). Estimation of Compensatory and Noncompensatory Multidimensional Item Response Models Using Markov Chain Monte Carlo. *Applied Psychological Measurement*, 27(6), 395-414. <https://doi.org/10.1177/0146621603258350>
- Bruin, J. (2011). *newtest: command to compute new test* @ONLINE. <https://stats.idre.ucla.edu/stata/ado/analysis/>
- Cabrera-Nguyen, E. (2010). Author Guidelines for Reporting Scale Development and Validation Results in the Journal of the Society for Social Work and Research. *Journal of the Society for Social Work and Research*, 1, 99-103. <https://doi.org/10.5243/jsswr.2010.8>
- Chalmers, R. P. (2012). mirt: A Multidimensional Item Response Theory Package for the R Environment. *Journal of Statistical Software*, 48(6), 1-29. <https://doi.org/10.18637/jss.v048.i06>
- Chalmers, R. P. (2016). Generating Adaptive and Non-Adaptive Test Interfaces for Multidimensional Item Response Theory Applications. *Journal of Statistical Software*, 71(5), 1-39. <https://doi.org/10.18637/jss.v071.i05>
- Factor Analysis Rotation*. (s.f.). Consultado el 16 de abril de 2021, desde <https://www.ibm.com/docs/en/spss-statistics/SaaS?topic=analysis-factor-rotation>
- Forina, M., Armanino, C., Lanteri, S. & Leardi, R. (1989). Methods of varimax rotation in factor analysis with applications in clinical and food chemistry. *Journal of Chemometrics*, 3(S1), 115-125. <https://doi.org/10.1002/cem.1180030504>
- Graham, J. W. (2009). Missing Data Analysis: Making It Work in the Real World. *Annual Review of Psychology*, 60(1), 549-576. <https://doi.org/10.1146/annurev.psych.58.110405.085530>
- Harris, D. (1989). Comparison of 1-, 2-, and 3-Parameter IRT Models. *Educational Measurement: Issues and Practice*, 8(1), 35-41. <https://doi.org/10.1111/j.1745-3992.1989.tb00313.x>
- Honaker, J., King, G. & Blackwell, M. (2011). Amelia II: A Program for Missing Data. *Journal of Statistical Software*, 45(7), 1-47. <http://www.jstatsoft.org/v45/i07/>
- Kang, T. & Chen, T. T. (2008). Performance of the Generalized S-X2 Item Fit Index for Polytomous IRT Models. *Journal of Educational Measurement*, 45(4), 391-406. <https://doi.org/10.1111/j.1745-3984.2008.00071.x>
- Kao, F. M. Y. S. T. (2014). Item response theory for measurement validity. *Shanghai archives of psychiatry*, 26(3), 171-177. <https://doi.org/10.3969/j.issn.1002-0829.2014.03.010>
- Kline, P. (2014). *An Easy Guide to Factor Analysis*. Routledge. <https://doi.org/10.4324/9781315788135>
- Liao, W.-W., Ho, R.-G., Yen, Y.-C. & Cheng, H.-C. (2012). The Four-Parameter Logistic Item Response Theory Model As a Robust Method of Estimating Ability Despite Aberrant Responses. *Social Behavior and Personality: an international journal*, 40(10), 1679-1694. <https://doi.org/10.2224/sbp.2012.40.10.1679>

- Little, R. J. A. (1988). A Test of Missing Completely at Random for Multivariate Data with Missing Values. *Journal of the American Statistical Association*, 83(404), 1198-1202. <https://doi.org/10.1080/01621459.1988.10478722>
- Little, R. J. A. & Rubin, D. B. (2002). *Statistical Analysis with Missing Data*. John Wiley & Sons, Inc. <https://doi.org/10.1002/9781119013563>
- Lloret-Segura, S., Ferreres-Traver, A., Hernández-Baeza, A. & Tomás-Marco, I. (2014). El análisis factorial exploratorio de los ítems: una guía práctica, revisada y actualizada. *Anales de Psicología*, 30(3). <https://doi.org/10.6018/analesps.30.3.199361>
- Lord, F. M. (2012). *Applications of Item Response Theory To Practical Testing Problems*. Routledge. <https://doi.org/10.4324/9780203056615> (Fecha inicial de publicación 1980)
- Mair, P. (2018). *Modern Psychometrics with R*. Springer International Publishing. <https://doi.org/10.1007/978-3-319-93177-7>
- Maydeu-Olivares, A. & Joe, H. (2005). Limited-and full-information estimation and goodness-of-fit testing in 2 n contingency tables: A unified framework. *Journal of the American Statistical Association*, 100(471), 1009-1020.
- Maydeu-Olivares, A. & Joe, H. (2014). Assessing approximate fit in categorical data analysis. *Multivariate Behavioral Research*, 49(4), 305-328. <https://doi.org/10.1080/00273171.2014.911075>
- Mills, C. N. & Stocking, M. L. (1996). Practical issues in Large-Scale Computerized Adaptive Testing. *Applied Measurement in Education*, 9(4), 287-304. https://doi.org/10.1207/s15324818ame0904_1
- mirt: Full-Information Item Factor Analysis (Multidimensional Item Response Theory). (s.f.). Consultado el 17 de abril de 2021, desde <https://www.rdocumentation.org/packages/mirt/versions/1.33.2/topics/mirt>
- Oppl, S., Reisinger, F., Eckmaier, A. & Helm, C. (1968). A flexible online platform for computerized adaptive testing. *International Journal of Educational Technology in Higher Education*, 14(1).
- Oppl, S., Reisinger, F., Eckmaier, A. & Helm, C. (2017). A flexible online platform for computerized adaptive testing. *International Journal of Educational Technology in Higher Education*, 14(1). <https://doi.org/10.1186/s41239-017-0039-0>
- Osborne, J. W. (2015). What is Rotating in Exploratory Factor Analysis? <https://doi.org/10.7275/HB2G-M060>
- R Core Team. (2019). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. Vienna, Austria. <https://www.R-project.org/>
- Rasch, G. (1960). *Studies in mathematical psychology: I. Probabilistic models for some intelligence and attainment tests*. Nielsen Lydiche. <https://www.routledge.com/Handbook-of-Item-Response-Theory-Volume-1-Models/Linden/p/book/9780367220013>
- Reckase, M. D. (2009). *Multidimensional Item Response Theory*. Springer New York. <https://doi.org/10.1007/978-0-387-89976-3>
- Rhoads, C. H. (2012). Problems with Tests of the Missingness Mechanism in Quantitative Policy Studies. *Statistics, Politics, and Policy*, 3(1). <https://doi.org/10.1515/2151-7509.1012>
- Rönkkö, M. (2020, 24 de junio). *Levels and patterns of missing data*[Archivo de vídeo]. Youtube. <https://www.youtube.com/watch?v=rmezzIK1Zd4>
- Roth, P. L. (1994). MISSING DATA: A CONCEPTUAL REVIEW FOR APPLIED PSYCHOLOGISTS. *Personnel Psychology*, 47(3), 537-560. <https://doi.org/10.1111/j.1744-6570.1994.tb01736.x>
- Rubin, D. B. (1976). Inference and missing data. *Biometrika*, 63(3), 581-592. <https://doi.org/10.1093/biomet/63.3.581>
- Schmitt, T. A. (2011). Current Methodological Considerations in Exploratory and Confirmatory Factor Analysis. *Journal of Psychoeducational Assessment*, 29(4), 304-321. <https://doi.org/10.1177/0734282911406653>
- Schreiber, J., Nora, A., Stage, F., Barlow, E. & King, J. (2006). Reporting Structural Equation Modeling and Confirmatory Factor Analysis Results: A Review. *Journal of Educational Research - J EDUC RES*, 99, 323-338. <https://doi.org/10.3200/JOER.99.6.323-338>
- Spray, J., Davey, T., Reckase, M., Ackerman, T. & Carlson, J. (1990). Comparison of Two Logistic Multidimensional Item Response Theory Models, 46.
- Sulis, I. & Porcu, M. (2017). Handling Missing Data in Item Response Theory. Assessing the Accuracy of a Multiple Imputation Procedure Based on Latent Class Analysis. *Journal of Classification*, 34(2), 327-359. <https://doi.org/10.1007/s00357-017-9220-3>

- Thompson, N. (2018, 25 de enero). *What is the Simpson-Hetter Item Exposure Control?* Consultado el 16 de abril de 2021, desde <https://assess.com/2018/01/25/simpson-hetter-item-exposure-control/>
- Vale, C. D. (2015). Computerized Item Banking. En S. Lane, M. R. Raymond & T. M. Haladyna (Eds.), *Handbook of Test Development* (2.^a ed., pp. 261-287). Routledge. <https://doi.org/10.4324/9780203102961> (Fecha inicial de publicación 2006)
- van der Linden, W. J. (2019). *Handbook of Item Response Theory: Volume 1: Models*. Routledge. <https://www.routledge.com/Handbook-of-Item-Response-Theory-Volume-1-Models/Linden/p/book/9780367220013>
- van Buuren, S. (2018). Flexible Imputation of Missing Data. Routledge. <https://stefvanbuuren.name/fimd/>
- van Buuren, S. & Groothuis-Oudshoorn, K. (2011). mice: Multivariate Imputation by Chained Equations in R. *Journal of Statistical Software*, 45(3), 1-67. <https://www.jstatsoft.org/v45/i03/>
- Velada, R., Caetano, A., Bates, R. & Holton, E. (2009). Learning transfer – validation of the learning transfer system inventory in Portugal. *Journal of European Industrial Training*, 33. <https://doi.org/10.1108/03090590910985390>
- Wang, C. & Chang, h.-h. (2011). Item Selection in Multidimensional Computerized Adaptive Testing—Gaining Information from Different Angles. *Psychometrika*, 76, 363-384. <https://doi.org/10.1007/s11336-011-9215-7>
- Warm's Mean Weighted Likelihood Estimates (WLE) of Rasch Measures. (s.f.). Consultado el 16 de abril de 2021, desde <https://www.rasch.org/rmt/rmt211h.htm>
- Worthington, R. L. & Whittaker, T. A. (2006). Scale Development Research: A Content Analysis and Recommendations for Best Practices. *The Counseling Psychologist*, 34(6), 806-838. <https://doi.org/10.1177/0011000006288127>
- Xu, J., Paek, I. & Xia, Y. (2017). Investigating the Behaviors of M2 and RMSEA2 in Fitting a Unidimensional Model to Multidimensional Data. *Applied Psychological Measurement*, 41(8), 632-644. <https://doi.org/10.1177/0146621617710464>

Apéndice

1. Apéndice A: Pequeño tutorial en R

Instalar R y RStudio

R está disponible para las plataformas **Windows**, **OS X** y **Linux/Unix** y la principal colección de recursos en línea es **Comprehensive R Archive Network** (CRAN). Si va al sitio web del proyecto R en <http://www.r-project.org/>, podrá navegar por las diferentes secciones del proyecto e informarse mejor de la labor de R Project. El primer parágrafo – *Getting Started* – conduce a los diferentes mirrors* que tiene la web. Una vez se selecciona el más cercano a su ubicación podrá seguir las indicaciones para descargar el archivo ejecutable e instalar R.

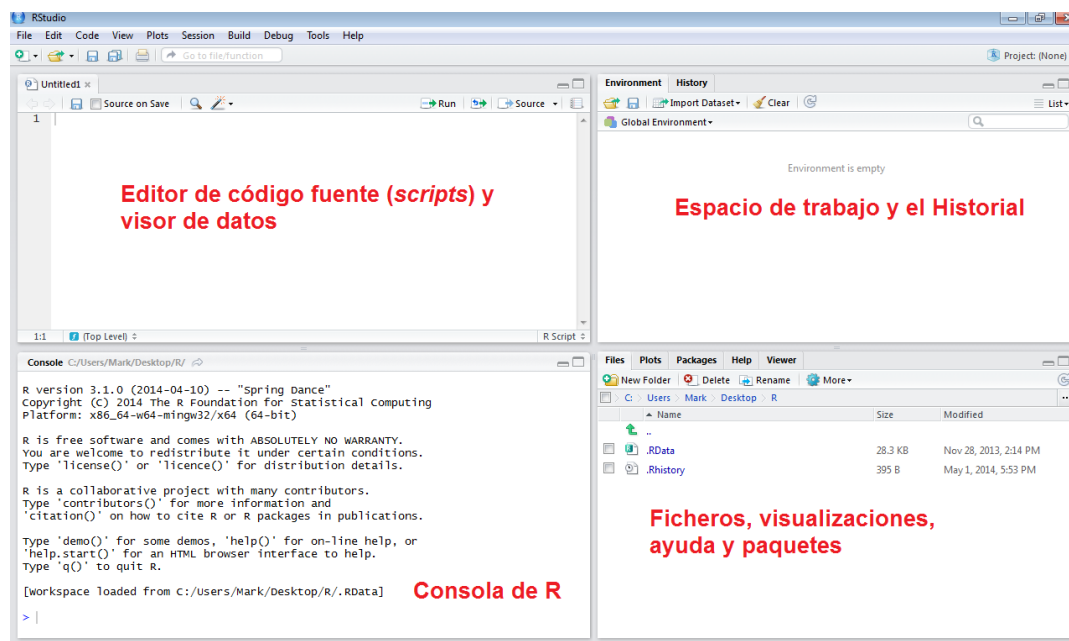
R es un lenguaje interpretado que es estrictamente sensible a las mayúsculas y los caracteres, por lo tanto, responde a instrucciones que siguen una sintaxis específica en una consola o una interfaz de comandos; el software interpreta y ejecuta ese código y devuelve el resultado.

Cuando se abre la aplicación de R se abre la interfaz que tiene por defecto. Tanto **OS X** como **Windows** tienen la interfaz sobria, cosa que podría intimidar a primera vista; por eso se propone **RStudio** como interfaz por defecto de este manual, aunque se pueda implementar en cualquier otra.

RStudio se descarga en su página oficial (<https://rstudio.com/products/rstudio/download/>) y es completamente gratuita. Tiene un aspecto mucho más depurado formado por cuatro componentes o paneles principales: la consola; el editor o script; el historial y el entorno, y por otro lado, los ficheros, gráficos y demás.

Figura 1

RStudio



Nota. En la imagen se observan los diferentes paneles de la interfaz gráfica de RStudio.

* CRAN no es una sola web, es una colección de servidores que contienen copias idénticas de toda la información de CRAN, también llamados mirros (espejos). Existen para agilizar la descarga de contenido en el lugar del mundo en el que estés, de esta forma, se reduce el tráfico de internet internacional o de larga distancia.

Aunque la funcionalidad de **RStudio** es muy extensa, lo más importante por ahora es que todo el código que se precie debería escribirse en script y no directamente en la consola. Al iniciar **RStudio** no hay ningún script abierto a menos que se tenga guardado uno; en File se encuentra todo lo necesario para crear script o proyectos nuevos. La interfaz es altamente personalizable, se puede elegir el color, la tipografía y la distribución de los paneles, entre otros, en *Tools* → *Global Options* → *Appearance*.

RStudio, al igual que cualquier otro editor de texto tiene comandos integrados que se pueden aprender con las **RStudio cheatsheet**, pero Ctrl+Enter sería el más común ya que ejecuta una línea de código en los script, seguido de Ctrl+Shift+S, que lee todo el código de un script sin mostrar ningún resultado en la consola; muy útil para tener en el entorno todas las funciones necesarias para el TAI de forma rápida.

Por defecto, la consola o intérprete que se ve en la Figura 1, indica si **R** está listo y esperando un comando con el símbolo >. Si se estuviese ejecutando un comando muy costoso, en la parte derecha superior de la consola aparecería una señal de Stop que detendría el proceso, pero si **RStudio** está consumiendo muchos recursos en ese momento puede ser que la sesión se bloquee y se pierda parte del trabajo sino se ha guardado, por eso se recomienda guardar el progreso antes de ejecutar un comando que se prevé costoso o probarlo con una porción de la base de datos original, entre otros.

Una sesión en **R** activa siempre tiene un directorio (carpeta) asociado, a menos que se especifique la ruta de un archivo en específico, de otro directorio y para saber en qué directorio se está trabajando use la función *getwd*.

Instalar y cargar paquetes

Con la instalación de R viene una lista pequeña de paquetes instalada, por lo tanto, solo necesitan ser cargados con la función *library*.

```
library(MASS)
```

Cargando paquetes o librerías, se puede acceder a todas las funciones y bases de datos que contenga la librería. Si no se cargasen, se tendría que especificar de qué paquete proviene cada función que se quiera utilizar.

```
MASS::fractions
```

Por otro lado, hay miles de paquetes disponibles en CRAN, que no se encuentran en la instalación base y que, por lo tanto, deben ser instalados manualmente. Para ello se necesita conexión a internet, ya que se descargarán de la mirror de la cual se haya descargado R; por esto es importante escoger el servidor local más cercano. Una vez instalado, ese paquete es como cualquier otro y, por lo tanto, también se debe cargar con *library*.

```
install.packages(mirt)
library(mirt)
```

Documentación y guardado

Hasta ahora ya se han mostrado algunas funciones: *library* se encarga de cargar librerías pero tiene muchos matices y opciones que por ahora, no sería lógico explicar. Siempre que se tenga alguna duda sobre qué opciones hay dentro de la función o cómo es una base de datos, se puede utilizar *help* o *?* y en el panel de ayuda se mostrará un resumen general de aquello que estemos buscando ayuda. Además, en el mismo código se pueden escribir comentarios si van precedidos de almohadillas.

```
# Esto es un comentario que no afecta al código
help(data.frame); ?lm
```

Cuando se quieran guardar los progresos en RStudio se debe prestar atención a dos cosas, cualquier nuevo objeto o función que se ha creado durante la sesión, que aparece en el panel de Environment y segundo, todos aquellos scripts que se hayan escrito. En cada nuevo proyecto (en la pestaña File se puede crear cualquier tipo de proyecto) de R se creará automáticamente un documento `.RData`^{**}, el cual guarda automáticamente el estado de la sesión antes de cerrar RStudio, por lo tanto, se mantienen todos los scripts y los objetos creados.

Datos

R es capaz de trabajar con los siguientes tipos de datos:

Tabla 1

Tipos de datos

Categoría	
Números	1, 2, 3...
Enteros	1L, 2L, 3L...
Números complejos	0 + 5i
Caracteres	a, b, c...
Valores lógicos	TRUE, FALSE

Nota. Los números equivalen a los números naturales \mathbb{N} y los enteros, también llamados así en matemáticas \mathbb{F} , son un tipo de número que no tiene decimales, en R se especifican con una L a la derecha y se diferencian principalmente de los números por la memoria que ocupan, los enteros ocupan menos. En R, el único número complejo es i , los caracteres se refieren a cualquier tipo de texto y los valores lógicos se usan para trabajar con operadores booleanos.

Cualquier dato puede asignarse a un objeto en la sesión en la que se esté trabajando, como, por ejemplo, el resultado de una suma. En R se pueden asignar de dos formas: usando una flecha (`<-`) o usando el signo igual (`=`).

```
entero <- 10L; suma = 12 + 3
entero; suma
```

```
[1] 10
[1] 123
```

Como se puede ver, R mostrará el valor asignado a un objeto cuando se ejecute en la consola. Además, cuando se usa un objeto en otras operaciones, R lo sustituye por el valor que tiene asignado.

```
suma / 5
```

```
[1] 3
```

^{**} Cada vez que se crea un proyecto se debe especificar el directorio donde se guardará todo lo que se derive del proyecto, es decir, todos los scripts, gráficos y el documento `.RData`, entre otros.

Estructuras de datos

Aunque R contemple más formas para guardar datos, aquí se van a mostrar las más básicas:

Tabla 2

Estructuras básicas

Objeto	Valores	¿Varios valores en el mismo objeto?
Vectores	números, caracteres, números complejos o valores lógicos	No
Factores	números o valores lógicos	No
Matrices	números, caracteres, números complejos o valores lógicos	No
Data frame	números, caracteres, números complejos o valores lógicos	Sí
Listas	números, caracteres, números complejos, valores lógicos, funciones, expresiones...	Sí

Un vector es una secuencia de elementos, que como se ve en la tabla puede contener varios tipos de datos; se crea con la función `c` que significa concatenar. Aunque se intenten concatenar valores diferentes R acabará forzándolos a que sean de solo un tipo, en este caso caracteres, ya que cada valor está entre comillas.

```
vector <- c(1 + 2, 3^3, TRUE, 'Hola')
vector
```

```
[1] "3"      "27"     "TRUE"   "Hola"
```

Los factores se crean con la función `factor` y es la mejor forma de contener datos categóricos. Automáticamente detecta qué grupos de datos categóricos hay en el vector y se muestran en Levels.

```
factor <- factor(c('Hombre', 'Mujer', 'Mujer', 'Desconocido'))
factor
```

```
[1] Hombre      Mujer      Mujer      Desconocido
Levels: Desconocido Hombre Mujer
```

Las matrices es una colección de datos organizados en una tabla de m filas por n columnas a la que se le pueden aplicar operaciones aritméticas, al igual que una matriz matemática. Aunque acepten varios tipos de valores, lo normal es que contengan valores numéricos.

```
matriz <- matrix(1:9, ncol = 3, nrow = 3)
matriz
```

```
      [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
```

Los `data.frame` son parecidos a las matrices, ya que también son tablas de m filas por n columnas pero está pensado para contener datos heterogéneos y se podría extrapolar su comportamiento a la de una tabla de Excel.


```
dataframe <- data.frame(números = 1:10, caracteres = alphabet[1:10], enteros = 1:10L, lógicos =
  sample(c(TRUE, FALSE), size = 10, replace = T))
```

	números	caracteres	integers	lógicos
1	1	a	1	FALSE
2	2	b	2	FALSE
3	3	c	3	TRUE
4	4	d	4	FALSE
5	5	e	5	TRUE

Finalmente, las listas pueden contener elementos de diferente tipo (matrices, data.frame, vectores, funciones, etc) e incluso contener listas dentro de listas.

```
lista <- list(vector = vector, factor = factor, matriz = matriz, datarframe = dataframe, lista =
  list(vector))
lista
```

```
$vector
[1] "3"      "27"     "TRUE"   "Hola"

$factor
[1] Hombre      Mujer      Mujer      Desconocido
Levels: Desconocido Hombre Mujer

$matriz
      [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9

$datarframe
      números caracteres integers lógicos
1          1          a          1     TRUE
2          2          b          2     TRUE
3          3          c          3    FALSE
4          4          d          4     TRUE
5          5          e          5     TRUE

$lista
$lista[[1]]
[1] "3"      "27"     "TRUE"   "Hola"
```

Indexar

Hay muchas formas de acceder a un dato, columna, componente, etc, de cualquiera de los objetos que se han explicado antes y a todos se puede acceder con corchetes. Los vectores y factores al tener solo una dimensión tendrán un índice o una secuencia (p. ej. 1:5) entre corchetes.

```
vector[1]; factor[c(1,4)]
```

```
[1] "3"
[1] Hombre      Desconocido
Levels: Desconocido Hombre Mujer
```


Las matrices y los data frames tendrán dos, el primero acorde a la fila y el segundo a la columna.

```
matriz[1,2]; dataframe[2,5]
```

```
[1] 4  
[1] b  
Levels: a b c d e
```

Las listas necesitan un índice también, al ser un objeto de una sola dimensión, pero para acceder a un componente o varios se deben utilizar dos corchetes, si no se hace así, se accede a un componente pero este no cambia de clase.

```
lista[[1]]; lista[[3]][1,1]; lista[3][2,3]; lista[[5]][[1]][2]
```

```
[1] "3"      "27"     "TRUE"  "Hola"  
[1] 1  
Error in lista[3][2, 3] : incorrect number of dimensions  
[1] "27"
```

A diferencia de los demás, a los data.frames o las listas también se puede acceder por el nombre que tenga la columna o el componente de la lista a través del signo del dólar.

```
dataframe$lógicos[c(1,2)]; lista$vector
```

```
[1] TRUE TRUE  
[1] "3"      "27"     "TRUE"  "Hola"
```

Funciones

Una función también es un objeto en R, pero este no sirve para contener información. Al igual que una función matemática, esta tiene unos parámetros (argumentos) a los que se les puede aplicar otras funciones y operaciones aritméticas, entre otros. Se definen por *function* y entre paréntesis se definen los argumentos y entre llaves se define el cuerpo de la función.

```
suma.columna <- function(datos, columna){  
  
  sum(datos[,columna])  
  
}  
  
suma.columna(matriz, 3)
```

```
[1] 24
```

2. Apéndice B: Funcionamiento del ítem

En el Capítulo 2 se han mostrado descripciones matemáticas sobre la interacción persona-ítem, cosa que conlleva a tener una forma poco intuitiva de entender las características de un ítem.

En esta sección se proporcionan otras formas estadísticas de describir el funcionamiento de los ítems basadas en los modelos compensatorios de la MIRT, porque tienen representaciones gráficas menos complejas que los parcialmente compensatorios.

Dificultad y discriminación del ítem

En la TRI, la discriminación del ítem está directamente relacionada con el parámetro a y equivale a la pendiente de la CCI y, más exactamente, la pendiente equivale a

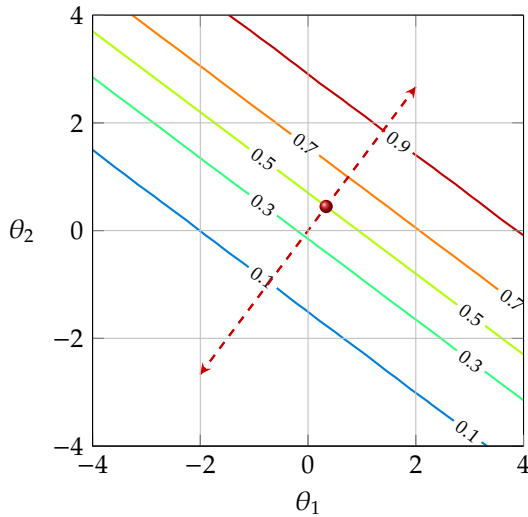
$$\frac{1}{4}a_i(1 - c_i) \quad (1)$$

Dicho de otro modo, la pendiente equivale a qué rapidez aumenta la probabilidad de acierto si aumenta el nivel de rasgo. Esta velocidad se puede observar en la Figura 2: cuánto más próximas están las líneas más discriminación tiene el ítem y, por tanto, más cambia la probabilidad de acierto si se cambia el valor de θ_1 o θ_2 .

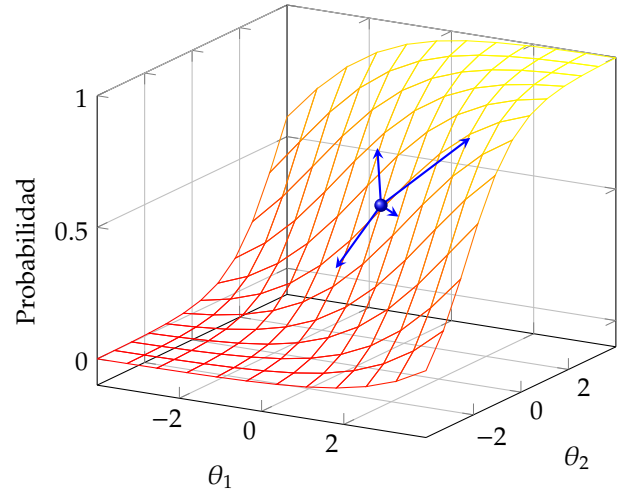
En la SCI no solo se define la pendiente por a , sino que también se debe evaluar la dirección en la que se mide. Si desde cualquier punto de la SCI se quiere ascender o descender lo más rápido posible por ella, solo hay una dirección que permita tal cosa; esta dirección es la que define la recta perpendicular a todas las líneas de probabilidad que se ve en la siguiente figura.

Figura 2

Recta de máxima pendiente



(a) *Nota.* Se puede apreciar que la recta roja es perpendicular a las líneas de probabilidad, además, se puede obtener el ángulo que tiene respecto a θ_1 y θ_2 y, en este caso, equivalen a 53.13° y 36.87° .



(b) *Nota.* El gráfico muestra diferentes direcciones en las que se podría evaluar la pendiente. Si se tuviese que llegar caminando desde el centro hasta la parte más elevada o llana de la CCI, solo habría una dirección que lo permitiría y esta es la de máxima pendiente.

La discriminación en la MIRT se llama MDISC y equivale a

$$MDISC_i = \sqrt{\sum_{k=1}^m a_{ik}^2} \quad (2)$$

Aunque no tenga una representación gráfica clara por sí sola, si se utiliza la ecuación 1.8 y se substituye a_i por MDISC se obtiene la velocidad exacta a la que aumenta la probabilidad de acierto cada vez que se avanza una unidad por la dirección de máxima pendiente, como se ve en la Figura 2.9. Por otro lado, los valores de MDISC no pueden ser negativos y cuanto más grande es su valor, más discriminatorio es el ítem.

Por lo que respecta a la dificultad, en la TRI esta equivale al valor de θ donde se encuentra el punto de inflexión – como se aprecia en la Figura 2.1 – que a su vez es el punto que define una probabilidad de acierto del 50 %. En la MIRT ese parámetro de dificultad se llama MDIFF y puede tener valores negativos, que indican ítems más difíciles, y valores positivos, que indican una mayor facilidad del ítem.

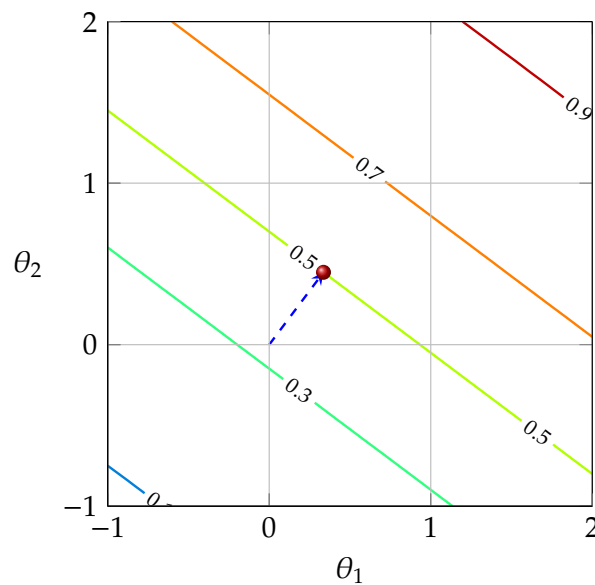
Tomando como referencia la Figura 8, MDIFF equivale a la distancia entre el origen $[0\ 0]$ y la esfera que interseca la recta de máxima pendiente y la probabilidad del 50 %. Esta distancia equivale a

$$MDIFF_i = \frac{-d_i}{\sqrt{\sum_{k=1}^m a_{ik}^2}} \quad (3)$$

La siguiente figura muestra una representación gráfica de MDIFF.

Figura 3

MDIFF



Nota. El valor de MDIFF de un ítem con $a_1 = .75$, $a_2 = 1$ y $d = -.7$ es 0.56 y equivale a la longitud de la recta azul.

Información del ítem

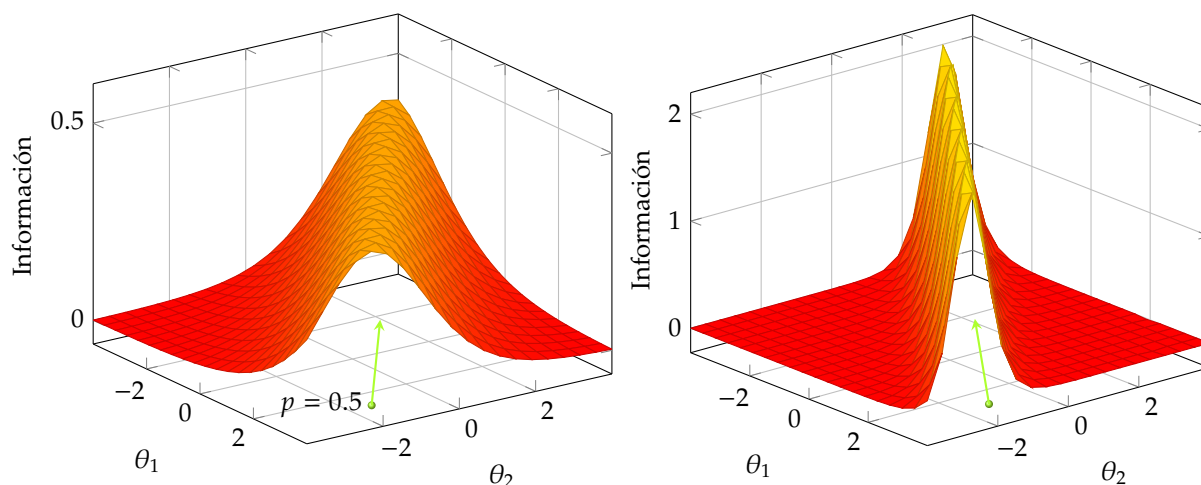
La información de un ítem, que se usa tanto en la TRI como en la MIRT, es un indicador estadístico que informa de la calidad de cualquier estimación del nivel de habilidad de una persona. Aunque se hayan hecho varias formulas para definirla (Reckase pag 47 Birnbaum Kendall Stuart), es más fácil entenderla como la derivada de la SCI o de la CCI.

Por ejemplo, la superficie de información de la Figura 8 se muestra a continuación y, visualmente, equivale a plegar la SCI sobre si misma de forma que quede una superficie simétrica.

Esta superficie puede ser más pronunciada dependiendo de la dirección en la que se evalúe la información, pero para saber la información exacta de un ítem en la MIRT, se tendría que evaluar en todas las direcciones.

Figura 4

Información del ítem



(a) *Nota.* La sección más elevada de la superficie de información tiene la misma dirección que las líneas de contorno de la Figura 8, además, se puede observar como los valores de θ de los que se tiene más información son aquellos que tienen una probabilidad de acierto del 50 % (flecha amarilla).

(b) *Nota.* El gráfico muestra la superficie de información para un ítem con $a_1 = 2.25$, $a_2 = 3$ y $d = -.7$. Se puede observar como la información aumenta para aquellas combinaciones de θ_1 y θ_2 cercanas a una probabilidad de acierto del 50 % al igual que en la Figura 2.8, pero la superficie ahora es más estrecha.

Además, la discriminación y la información están estrechamente relacionadas: la SCI es más pronunciada cuanto más lo es la superficie de información y viceversa. Esto conlleva que más discriminación reduzca el número de los valores de θ más informativo, ya que la amplitud de la superficie es menor, como se observa en la Figura 2.10.

3. Apéndice C: Optimización

En este apartado se propone un ejemplo práctico basado en estimar el nivel de habilidad de una persona a través de máxima verosimilitud. De esta forma se quiere ilustrar el concepto de optimización matemática, basado en elegir el mejor conjunto de elementos que de respuesta a un tipo de problema matemático, y así para extrapolar cómo la función *mirt* llega a una solución.

En la sección de Imputación del Capítulo 1 se explica cómo funciona el algoritmo esperanza maximización (EM), que en sí mismo, también es un proceso de optimización, pero aquí se va a explicar la optimización por máxima verosimilitud (MLE) des de una perspectiva más matemática, para estimar la habilidad de una persona.

MLE se definiría como:

El criterio por máxima verosimilitud se basa en estimar aquellos valores de θ que resultan más probables según una distribución de probabilidad, a través de maximizar su función de verosimilitud.

El proceso consta de dos aspectos principales: una función, cuyos parámetros son desconocidos y se quieren obtener aproximándolos a una solución aceptable, y una función objetivo que mide la calidad de los resultados que se van obteniendo durante el proceso de optimización, hasta llegar a una solución aceptable.

Estimación de habilidad

Este tipo de cálculos se hacen una vez la persona evaluada contesta a una serie de ítems, por lo tanto, se obtendría un vector de unos (correcto) y ceros (incorrecto).

La probabilidad de acertar un ítem es un evento discreto, es decir, se puede responder correcta o incorrectamente, por lo tanto, la distribución de probabilidad que se ajusta mejor al vector de respuestas de una persona evaluada es la distribución de Bernoulli.

Bernoulli

La distribución de Bernoulli se define por

$$f(x) = p^x(1 - p)^{n-x}$$

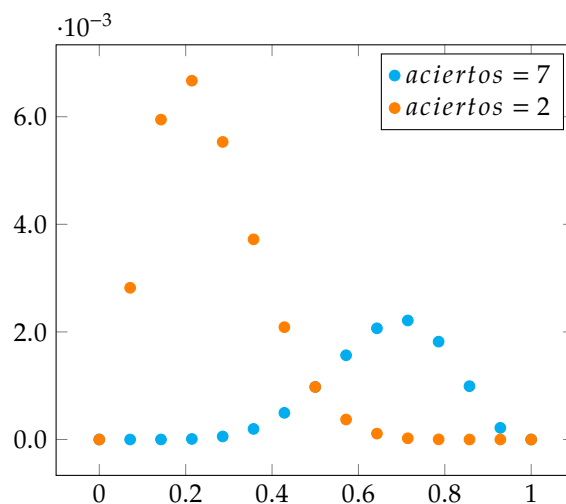
donde

- p es la probabilidad de un acierto.
- n es el número total de intentos.
- x es el número total de aciertos.

Si una persona contestase correctamente 7 preguntas de 10 muestra una probabilidad del 70 % de acertar un ítem, pero, se puede saber cómo de segura es esa probabilidad. Si se ponen a prueba diferentes probabilidades manteniendo fijo el número de aciertos se observa la siguiente distribución de probabilidad:

Figura 5

Distribución de Bernoulli



Nota. El gráfico muestra la verosimilitud para diferentes probabilidades. No es necesario establecer una distribución de probabilidad sobre lo que sería una simple media aritmética, pero como se ve más adelante, es importante entender el comportamiento de esta función.

Si se tuviesen el siguiente vector de respuestas para los siguientes ítems:

Tabla 3

Ítems de muestra

Parámetros			Respuestas
a_i	b_i	d_i	
0.8	0.5	-0.6	1
0.2	1.1	-1.2	0
1	0.2	-2	0

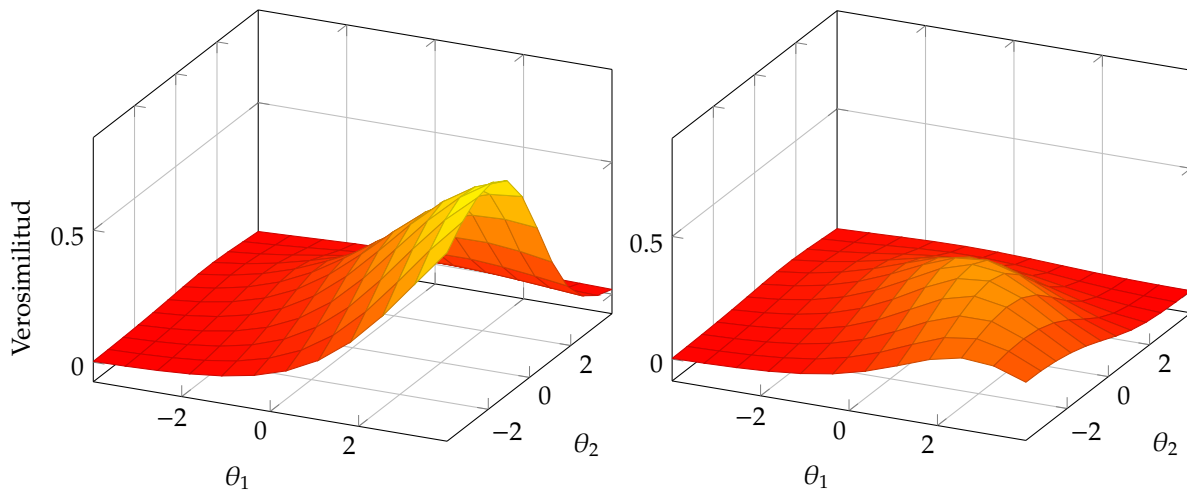
Bernuolli se vería como la siguiente ecuación si se substituyese p por la ecuación de la superficie característica del ítem (SCI) en la ecuación 2.5.

$$\left(\frac{e^{0.8\theta_1 + 0.5\theta_2 - 0.6}}{1 + e^{0.8\theta_1 + 0.5\theta_2 - 0.6}} \right) \left(1 - \left(\frac{e^{0.2\theta_1 + 1.1\theta_2 - 1.2}}{1 + e^{0.2\theta_1 + 1.1\theta_2 - 1.2}} \right) \right) \left(1 - \left(\frac{e^{1\theta_1 + 2\theta_2 - 2}}{1 + e^{1\theta_1 + 2\theta_2 - 2}} \right) \right)$$

En este punto si se substituye θ_1 y θ_2 por varios valores entre -4 y 4, obtendríamos una superficie (gradiente) que determina cómo de probable es observar θ_1 y θ_2 , y así obtener el nivel de habilidad de una persona.

Figura 6

Máxima verosimilitud



(a) Nota. Aquí se muestra la superficie para los dos primeros ítems. Se observa como no hay un valor máximo claro, ya que se necesitarían más valores de θ para determinarlo. En este punto, la función tiende a infinito para cada nivel de θ .

(b) Nota. Aquí se observa un máximo alrededor de $\theta_1 = 1.6$ y $\theta_2 = -0.8$. Además el valor en el eje de ordenadas es más bajo y esto se ocurre porque se van multiplicando más probabilidades a medida que se contestan más ítems.

El estimador de máxima verosimilitud puede dar como válido un valor de θ imposible o no finito para combinaciones de ítems muy diferentes y pocas respuestas o respuestas improbables. Esto suele ser síntoma de un banco de ítems mal calibrados o poco sensible a determinados niveles de rasgo, ya que en condiciones normales este estimador da buenos resultados; su precisión aumenta a medida que se administran más ítems. Cabe destacar, que el proceso de optimización de la función *mirt* o de cualquier otro no buscará un máximo (o un mínimo) probando todos los valores de θ posibles, ya que sería, en algunos casos, imposible a nivel computacional. Para ello se utilizan algoritmos como el de Newton-Raphson, que se aproximan a una solución, minimizando los costes computacionales.

4. Apéndice D: Seleccionar el próximo ítem del banco de ítems

Maximizar la información en la dirección de menor información

Este método propone que, a medida que se va estimando el nivel de habilidad, se trie el siguiente ítem que aporta más información en la dirección de menor información que tienen los ítems ya respondidos.

Como se ha visto en el Apéndice B la cantidad de información de un ítem, o varios, depende de la dirección que se elija des de cualquier combinación de θ . Si un ítem no discrimina en una dimensión, es decir, tiene un a_i igual a 0, no será útil para medir la habilidad en esa dimensión.

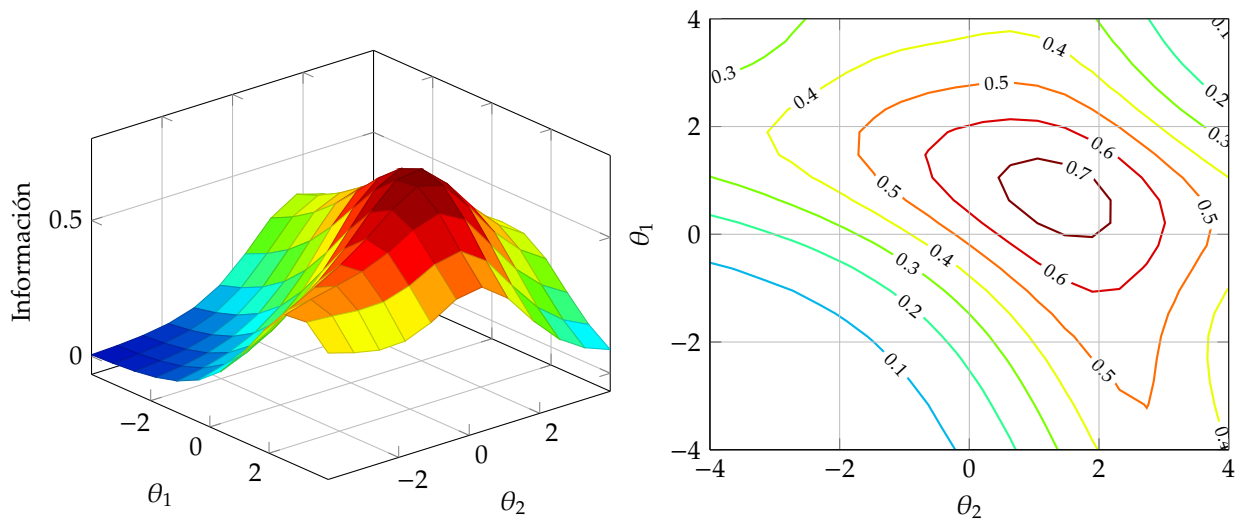
El proceso es relativamente sencillo y consta de los siguientes pasos:

1. Obtener la superficie de información para los ítems administrados.
2. Evaluar en qué dirección se tiene menos información.
3. Obtener la superficie de cada ítem, no administrado, para la dirección obtenida en el paso anterior.
4. Seleccionar el ítem con más información en el paso anterior.

Por ejemplo, si se hubiesen administrado los ítems de la Tabla 3 se tendría la siguiente superficie de información:

Figura 7

Información de ítems administrados



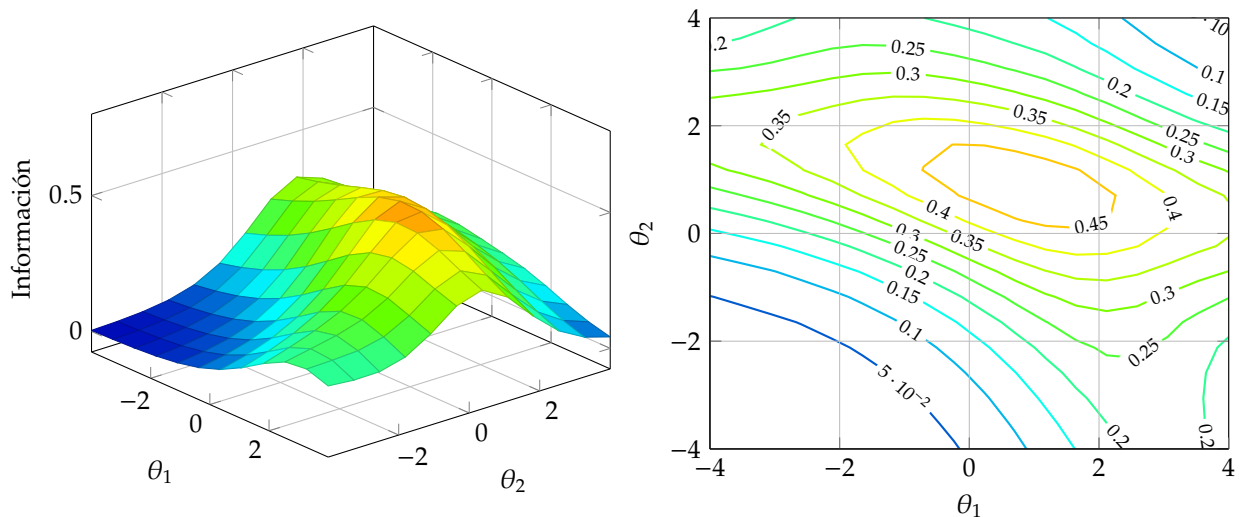
(a) *Nota.* La información de los ítems se ha obtenido teniendo en cuenta la dirección de máxima información para cada uno, es decir, esta es la máxima información que presentan los tres ítems.

(b) *Nota.* Los valores de θ con más información son aproximadamente $\theta_1 = 1$ y $\theta_2 = 1.5$. Además, los valores de θ negativos son los que necesitan más información, es decir, se necesita administrar ítems sensibles a niveles de θ bajos para tener más precisión sobre estos valores.

En el paso 2, se suma la información de cada ítem con la misma dirección para cada uno. Aunque teóricamente se deban evaluar todas las direcciones posibles, en la práctica, aquellas que estén muy próximas no darán resultados muy diferentes. Por lo tanto, si se evalúan ángulos de 0 a 90 grados, divididos cada 10 grados, la dirección de menor información es de 70° para θ_1 y 30° para θ_2 , como se muestra en la Figura 8. En el paso 3 esta dirección se evalúa para cada ítem que no se haya administrado y se selecciona el más informativo en el paso 4.

Figura 8

Información de ítems en dirección de menor información



Nota. Los gráficos muestran como la información global se reduce drásticamente en comparación a la Figura 4.

5. Apéndice E: Código

```
##### DATOS FALTANTES #####
## PATRONES
patrones(df)
## INFLUX OUTFLUX
plotflux(df)

##### IMPUTAR DATOS, AMELIA #####
df <- amelia(df, noms = 1:60, m = 1)$imputations$impl

##### OBTENCIÓN DE PARÁMETROS #####
n_items <- ncol(df)
mirtCluster(parallel::detectCores()-1)

# MODELO EXPLORATORIO
mirt.exp.3F <- mirt(df, 3, itemtype = '2PL', method = 'MHRM')
mirt.exp.2F <- mirt(df, 2, itemtype = '2PL', method = 'MHRM')
coef(mirt.exp.3F)

# MODELO CONFIRMATORIO
modelo.TwoTier.3F <- mirt.model('
    F1 = 1-60
    F2 = 1-60
    F3 = 1-60
    COV = F1*F2*F3
```



```

        ')
mirt.TwoTier.3F <- bfactor(df,
                        c(rep(1:4, each = 15)),
                        modelo.TwoTier.3F,
                        itemtype='2PL')
mirt.bifactor.4F <- bfactor(df,
                        c(rep(1:4, each = 15)),
                        itemtype='2PL')
mirtCluster(remove = TRUE)
coef(mirt.TwoTier.3F)

##### BONDAD DE AJUSTE #####
# ANOVA
anova.exp.1 <- anova(mirt.exp.2F, mirt.exp.3F)
anova.exp.1
anova.conf.1 <- anova(mirt.bifactor.4F, mirt.TwoTier.3F)
anova.conf.1

# itemfit
itemfit.mirt.exp.3F <- itemfit(mirt.exp.3F, QMC = T)
itemfit.mirt.exp.3F

# M2
M2.mirt.exp.3F <- M2(mirt.exp.3F, QMC = T)
M2.mirt.exp.3F

# residuals
residuals.mirt.exp.3F <- residuals(mirt.exp.3F, df.p = T, QMC = T)
cbind(which(residuals.mirt.exp.3F$df.p < 0.05, arr.ind = T),
      p = residuals.mirt.exp.3F$LD[which(residuals.mirt.exp.3F$df.p < 0.05)])

# clusters
parametros.mirt.exp.3F <- CoefToDataframe(modelo = mirt.exp.3F)
coordenadas.mirt.exp.3F <- polar_coords(parametros.mirt.exp.3F[,1:3],
                                       parametros.mirt.exp.3F$d)
Ward.mirt.exp.3F <- ward_mat(coordenadas.mirt.exp.3F[c('ang1', 'ang2', 'ang3')])
clusters.mirt.exp.3F <- hclust(as.dist(Ward.mirt.exp.3F), method = 'ward.D2')
plot(clusters.mirt.exp.3F)
rect.hclust(clusters.mirt.exp.3F, k = 3)

##### TAI #####
# Definir df
enunciados <- sprintf('Enunciado.%d', seq_len(n_items))
opciones <- matrix(c('Verdadero', 'Falso'), nrow = n_items, ncol = 2, byrow = T)
respuestas <- sample(c('Verdadero', 'Falso'), n_items, replace = T)
tipo <- c('radio')
df <- data.frame(Questions = enunciados,
                Option = opciones,
                Answer = respuestas,
                Type = tipo,
                stringsAsFactors = F)

# Definir preCAT
preTAI <- list(max_items = 5,
              criteria = 'Drule',
              method = 'ML')

# Definir design

```

```

contenido <- rep(c('ComprensiónLectora', 'ComprensiónOral',
                  'Ortografía', 'Cohesión'), each = 15)
contenido_prop <- c(ComprensiónLectora = 0.25,
                  ComprensiónOral = 0.25,
                  Ortografía = 0.30,
                  Cohesión = 0.20)
diseño <- list(min_SEM = 0.2,
              max_items = 50,
              content = contenido,
              content_prop = contenido_prop,
              max_time = 3600,
              exposure = 0.80)

# Definir ShinyGUI
GUI <- list(title = 'TAI: Ejemplo 1',
            authors = 'Carlos Pérez',
            instructions = c('Para contestar a un ítem se debe seleccionar siempre una de las dos
                             opciones,                               no se permiten respuestas en blanco y finalizará en una hora.', '
                             Siguiente ítem'))

# Definir mirtCAT
TAI <- mirtCAT(df = df, mirt2PL.coef, method = 'ML', criteria = 'Drule',
              preCAT = preTAI, design = diseño, shinyGUI = GUI)

# Visualizar y obtener el nivel de habilidad
summary(TAI)
plot(TAI)

```

6. Apéndice F: Funciones

patrones

```

patrones <- function(data){
  matrixplot(
    (matrix(colMeans(is.na(data))*1.8,
            nrow(data), ncol(data),
            byrow = T) + ifelse(is.na(data), NA, 0)),
    cex.lab = 1.5,
    cex.axis=1.5
  )
}

```

plotflux

```

plotflux <- function(data) {
  pat <- md.pairs(data)
  outflux <- rowSums(pat$rm) / (rowSums(pat$rm + pat$mm))
  influx <- rowSums(pat$mr) / (rowSums(pat$mr + pat$rr))
  dataframe <- data.frame(influx = influx, outflux = outflux, items = colnames(data),
                          Imputable = colMeans(data, na.rm = T) %in% 0:1)
  ggplot(dataframe, aes(influx, outflux, label = items, col = Imputable)) +
    geom_segment(aes(x = 0, y = 1, xend = 1, yend = 0)) +
    coord_cartesian(xlim = c(0:1),
                    ylim = c(0:1)) +
    theme_minimal() +
    geom_point() +

```

```

    scale_color_manual(values = c('black','red'))+
    geom_text_repel(
      max.overlaps = ncol(data),
      data = subset(datframe, influx < .5 & outflux < .5 | Imputable),
      segment.linetype = 3.5,
      family = 'mono',
      fontface = 'bold',
      size = 6) +
    theme(legend.position = 'none',
          axis.text=element_text(size=12),
          axis.title=element_text(size=14,face="bold"))
}

```

polar_coords

MDIFF

```

MDIFF <- function(a, d){
  -d / sqrt(sum(a^2))
}

```

MDISC

```

MDISC <- function(a){
  sqrt(sum(a^2))
}

```

interseccion2d

```

interseccion2d <- function(a1, a2, d){
  a <- list(c(a1, a2))
  angulo <- acos(a1/sqrt(a1^2 + a2^2))
  x0 <- cos(angulo) * mapply(MDIFF, a, d)
  y0 <- sin(angulo) * mapply(MDIFF, a, d)
  return(c(x0, y0, angulo / pi*180))
}

```

polar_coords

```
polar_coords <- function(a, d){  
  MMDD <- sapply(1:nrow(a), function(x){  
    c(MDIFF(a[x,],d[x]), MDISC(a[x,]))  
  })  
  if (ncol(a) == 2){  
    coords <- mapply(intrscion2d, a[,1],a[,2], d)  
    x1 <- MMDD[2,] * .25 * cos(coords[3,] * pi / 180) + coords[1,]  
    y1 <- MMDD[2,] * .25 * sin(coords[3,] * pi / 180) + coords[2,]  
    data.frame(a, d, MDIFF = MMDD[1,], MDISC = MMDD[2,], angl = coords[3,],  
              ang2 = 90 - coords[3,], x0 = coords[1,], y0 = coords[2,], x1, y1)  
  } else {  
    angulos <- t(apply(a, 1, function(x){  
      sapply(1:ncol(a), function(y) {  
        acos(x[y]/sqrt(sum(x^2))) / pi*180  
      })  
    })  
    )  
    colnames(angulos) <- sprintf("ang%d", 1:ncol(angulos))  
    data.frame(a, d, MDIFF = MMDD[1,], MDISC = MMDD[2,], angulos)  
  }  
}
```