PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

ESCUELA DE POSGRADO

MODELO DE DIAGNÓSTICO COGNITIVO LONGITUDINAL CON ESTRUCTURA JERÁRQUICA DE ORDEN SUPERIOR Y ATRIBUTOS DEPENDIENTES

Tesis para optar el grado académico de Maestro en Estadística
que presenta:

Cesar Manuel Villanueva Valerio

Asesor:

Luis Hilmar Valdivieso Serrano

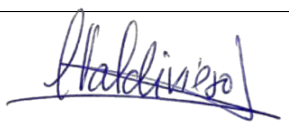
Lima, 2024

Informe de Similitud

Yo Luis Hilmar Valdivieso Serrano docente de la Escuela de Posgrado de la Pontificia Universidad Católica del Perú, asesor de la tesis titulada "Modelo de Diagnóstico Cognitivo Longitudinal con Estructura Jerárquica de Orden Superior y Atributos Dependientes", del autor Villanueva Valerio, César Manuel dejo constancia de lo siguiente:

- El mencionado documento tiene un índice de puntuación de similitud de 4 %. Así lo consigna el reporte de similitud emitido por el software Turnitin el 23/08/2024.
- He revisado con detalle dicho reporte y confirmo que cada una de las coincidencias detectadas no constituyen plagio alguno.
- Las citas a otros autores y sus respectivas referencias cumplen con las pautas académicas.

Lima, 02 de Setiembre de 2024

Apellidos y nombres del asesor: Valdivieso Serrano, Luis Hilmar	
DNI: 07958730 ORCID: https://orcid.org/0000-0002-8975-7557	Firma: 

Resumen

Diagnosticar el estado de aprendizaje de los estudiantes y determinar las habilidades subyacentes que permitan la comprensión de ciertos temas representan un desafío complejo en el ámbito educativo. Existen numerosos factores, tanto generales como específicos, que pueden influir en el desempeño individual para dominar dichas habilidades. Además, para hacerlo más desafiante, estas podrían estar interrelacionadas, formando una jerarquía en donde unas son pre requisito para acceder a otras más avanzadas.

Para abordar esta complejidad, se han desarrollado modelos de diagnóstico cognitivo que permiten construir perfiles detallados de las fortalezas y debilidades de los estudiantes en relación con habilidades específicas. Estos perfiles facilitan la creación de trayectorias de aprendizaje personalizadas, diseñadas para guiar a cada estudiante hacia el dominio de los conocimientos requeridos.

Las trayectorias de aprendizaje representan secuencias de habilidades que los estudiantes deben adquirir para alcanzar un objetivo educativo determinado. Estas trayectorias son dinámicas y requieren una evaluación continua para garantizar que se ajusten a las necesidades individuales de cada estudiante. En este sentido, resulta fundamental contar con modelos de diagnóstico cognitivo que sean capaces de adaptarse a los nuevos requerimientos educativos y proporcionar información precisa sobre el progreso de los estudiantes.

En este estudio, se analizarán dos modelos de diagnóstico cognitivo longitudinal de orden superior secuencial de reciente desarrollo. A través de un ejercicio de simulación y una aplicación con datos reales de una prueba matemática, se evaluará el desempeño y la capacidad clasificadora de estos modelos. Esta investigación contribuirá con la difusión de esta clase de modelos para promover su uso en los procesos de aprendizaje.

Palabras-claves: diagnóstico cognitivo, modelo de diagnóstico cognitivo jerárquico secuencial de orden superior, estados de conocimiento, atributos latentes.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

La educación es un derecho humano fundamental que permite una vida más digna, con menos desigualdad y mayor calidad según la Unesco. Es un problema complejo que requiere de herramientas precisas y adaptables a los diferentes contextos de las personas, sin importar su lugar de nacimiento, edad, lengua materna, cultura, entre otras dimensiones humanas.

Uno de los aspectos más importantes para garantizar la buena calidad de la educación es el diseño de procesos de aprendizajes adecuados considerando la capacidad cognitiva de los estudiantes y la dificultad de las habilidades requeridas para dominar los temas y sus interdependencias. Para ayudar a superar estos desafíos, los Modelos de Diagnóstico Cognitivo permiten a los profesionales identificar y comprender las debilidades más significativas del aprendizaje. A partir de este diagnóstico, se pueden diseñar currículos de estudios que garanticen mayores probabilidades de dominar conocimientos básicos y necesarios para el desarrollo de las personas. Por lo que, el uso de estas herramientas estadísticas son fundamentales para los profesionales y expertos de la educación pues los ayuda a medir el estado de desarrollo cognitivo de los estudiantes e identificar aquellas capacidades o habilidades que facilitan y potencian su aprendizaje.

En esta investigación, se plantearán dos modelos de diagnóstico cognitivo jerárquicos longitudinales en la que los atributos latentes subyacentes podrían interrelacionarse secuencialmente. Para ello, se estudiará la propuesta que estos atributos latentes, que explican el dominio de una habilidad o capacidad, se encuentran estructurados de manera jerárquica y, posiblemente, interconectada; por lo que, se incluirá en el modelamiento una relación de dependencia entre estos.

Se han desarrollado en la literatura, durante los últimos años, los Modelos de Diagnóstico

Cognitivo Longitudinal, o MDCL, (Pan et al. (2020); Zhan (2020)) que evalúan el dominio del *individuo* de ciertos *atributos latentes* a través de la resolución de *ítems* en una evaluación e identifican sus fortalezas y debilidades dentro de un periodo de tiempo. Los MDCL mejoran la capacidad predictiva de los tradicionales Modelos de Diagnóstico Cognitivos (MDC o CDM por sus siglas en inglés). De esta manera, los MDCL buscan proveer oportunamente de retroalimentación a los estudiantes para corregir e impulsar su aprendizaje (Zhan (2020)).

Un constructo latente, en adelante, "atributo", es una habilidad, destreza o conocimiento sobre un tema o campo no expresamente observable en la unidad examinada, que en adelante identificaremos como "individuo", y cuyo dominio es requerido para resolver correctamente uno o más ítems evaluados. Asimismo, se considera un "ítem", a cada pregunta, ejercicio o problema que requiere ser solucionado por un individuo durante la evaluación.

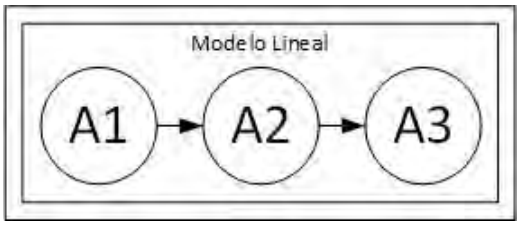
Los MDCL pueden ser de dos tipos principales. Por un lado, se encuentran los *Modelos basados en el análisis de transiciones de clases latentes* (Kaya y Leite (2017); Li et al. (2015)), en inglés "latent transition analysis-based models" que estudian las probabilidades de cambiar de una clase latente a otra o de mantenerse en la misma. Por el otro lado, están los *modelos estructurales latentes basados en órdenes superiores* (en inglés "Higher-Order latent structural-based models") (Lin et al. (2020); Lee (2017)) que, en una jerarquía de atributos agrupada por niveles, consideran el cambio de las variables latentes de niveles superiores ("órdenes superiores"), durante un tiempo definido, para inferir los cambios que podrían producir en los atributos de niveles inferiores (Lee (2017)).

La mayoría de MDCL desarrollados presuponen que los atributos latentes son independientes entre sí, por lo que ninguno requiere de otro para ser dominado. Esta forma de interpretar la realidad no es muy lógica, pues es natural asumir que existen atributos que pueden requerirse para entender otros [1]. Esta falta de dependencia entre los atributos genera restricciones para modelar ciertos conocimientos. Debido a esto, no se puede considerar la influencia de dominar unos y otros no en el diagnóstico del conocimiento del individuo sobre ciertos temas, ni su desempeño total.

Para superar esta restricción de independencia, Zhan y He (2021) propusieron un nuevo modelo Jerárquico de Diagnóstico Cognitivo de Orden Superior Secuenciales. Este considera

[1] Consideremos, por ejemplo, que tenemos dos atributos: Atr1 = "Comprender el contenido de un párrafo", y Atr2 = "Identificar la idea principal en un párrafo". Para realizar correctamente Atr2, es lógico suponer que primero se debe dominar Atr1; de lo contrario, se esperaría que el individuo falle al resolver algún ejercicio que evalúe Atr2.

Figura 1.1: Modelo secuencial lineal de atributos jerárquicos



Tomado de: Elaboración propia.

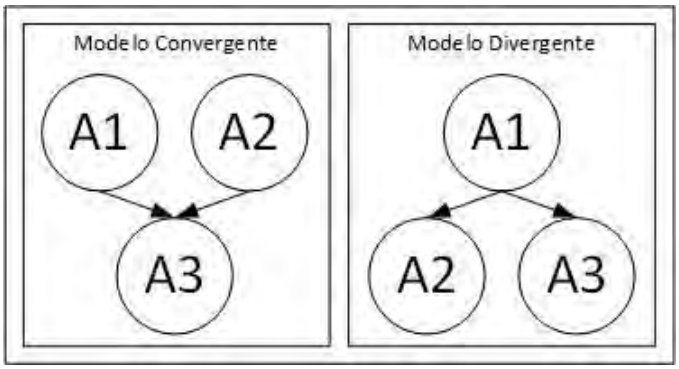
la existencia de una estructura jerárquica por niveles donde los atributos de niveles superiores (llamados de orden superior) influyen sobre los de niveles inferiores (llamados de orden inferior). Además, modela las relaciones de dependencia entre los atributos de un mismo nivel a través de una secuencia ordenada, donde se deben dominar los primeros atributos de la secuencia para poder acceder al dominio de los siguientes.

En la Figura 1.1, se aprecia el esquema de una jerarquía de 3 atributos secuenciales: A1, A2, A3. La relación de dependencia se indica mediante la flecha que los conecta: el atributo a donde apunta la flecha es el atributo que depende del que se encuentra al inicio de la flecha. A modo de ejemplo, podríamos tomar en cuenta los atributos: A1="Suma de números reales", A2="Multiplicación de números reales", y A3="Exponenciación de números reales".

El MDCL Jerárquico de Orden Superior Secuenciales, mediante este esquema de dependencias, indica que A1 es requerido para dominar A2 y, del mismo modo, sucede para A2 y A3. De este modo, este modelamiento permite calcular más precisamente cómo impacta el dominio o el no dominio de A1 sobre resolver ítems que requieran A2 y el desempeño general del individuo en la evaluación.

Asimismo, podemos formular más relaciones secuenciales. Por ejemplo, podríamos asumir que dos atributos, o más, son necesarios para dominar uno tercero y viceversa (donde ese tercer atributo permite el dominio de los dos primeros). Este es el caso de las jerarquías convergente y divergente respectivamente, los cuales pueden ser observados en la Figura 1.2.

Figura 1.2: Modelos secuenciales jerárquicos de atributos convergentes y divergentes



Tomado de: Elaboración propia.

1.1. Objetivos de la Tesis

El objetivo principal de esta tesis reside en el estudio de un modelo longitudinal de diagnóstico cognitivo que incorpora una estructura jerárquica latente de orden superior. Este modelo se caracteriza por atributos dependientes que posibilitan la estimación del nivel cognitivo de los individuos y la identificación de la trayectoria óptima para el desarrollo de su aprendizaje.

Los objetivos específicos de la tesis son los siguientes:

- Estudiar las propiedades de este modelo.
- Realizar estudios de simulación para evaluar la estimación de los parámetros del modelo.
- Comparar el desempeño del modelo frente a diferentes factores de análisis
- Aplicar el modelo a un conjunto de datos reales comparando este con otros modelos existentes.

1.2. Organización del trabajo

En el capítulo 2, se presenta el marco teórico que se usará para desarrollar los siguientes temas de la tesis. En este, se comienza identificando al problema de la tesis desde los campos de la psicometría y psicología cognitiva. Luego, veremos como este campo se combina con la estadística clásica y bayesiana para proponer modelos más sofisticados. Asimismo, se describen los modelos de diagnóstico cognitivo, base de los modelos longitudinales que se estudiarán en este trabajo.

En el capítulo 3, se desarrollará el modelo propuesto en esta tesis. Se explicará su origen y las fórmulas que lo sustentan, el significado de sus elementos (parámetros y coeficientes), sus propiedades y se estudiará la estimación bayesiana del modelo.

En el capítulo 4, realizaremos un estudio de simulación de recuperación de parámetros considerando tres estructuras jerárquicas y los dos modelos presentados para identificar.

En el capítulo 5, se realizará la aplicación empírica de los modelos presentados usando datos longitudinales de una evaluación sobre números racionales y operaciones en un grupo de estudiantes. Estos datos serán ajustados a los dos modelos propuestos en esta tesis.

Finalmente, en el capítulo 6, se brinda una síntesis de los resultados obtenidos en este trabajo. Asimismo, se propondrán preguntas que la presente investigación no ha respondido por escapar a su alcance, pero que podrían ser los siguientes pasos en esta línea de investigación.

Capítulo 2

Marco teórico

2.1. Psicometría y Estadística

Los métodos psicométricos y la psicología cognitiva tienen muchos desafíos en común en el campo educativo. Uno de los más complejos consiste en comprender y mejorar la capacidad de aprendizaje de las personas. Mientras que los primeros se centran en la medición y análisis de varias características psicológicas humanas, como capacidades y creencias; el segundo estudia diferentes dimensiones del procesamiento de la información, como la memoria, resolución de problemas, entre otros (Novick (1980)).

El trabajo psicométrico decanta en diferentes resultados, por ejemplo, en modelos matemáticos de variables latentes. Uno de los casos más conocidos y antiguos fue el trabajo de Spearman (1904) sobre Inteligencia General y su diferencia con habilidades determinadas, como la habilidad matemática. Este trabajo fue pionero en el uso de factores analíticos, que es un tipo de modelo de clases latentes usado aún más de cien años después con más sofisticación, para estudiar la capacidad intelectual humana y proponer una generalización estadística de sus hallazgos. Desde ese momento, el modelamiento de variables latentes ha generado una gran colección de modelos y técnicas que actualmente son usados para investigaciones de diferentes campos: psicológicos, biológicos, económicos, entre otras (Cai (2012)).

Para explicar cómo ambos campos se fueron integrando con el fin de potenciar el proceso de aprendizaje de las personas, debemos ir hacia atrás y presentar los conceptos más relevantes que permitieron la construcción de los modelos cognitivos que estudiaremos.

2.2. Variables Latentes

Las variables latentes son variables aleatorias no observables, cuya medición actual no es viable. Sin embargo, su introducción en un modelo permite dos grandes beneficios. En primer lugar, reduce la dimensionalidad del análisis con una pérdida pequeña de información. En segundo lugar, tangibiliza conceptos abstractos (como felicidad, calidad de vida, ansiedad) para su evaluación, aun cuando no existan instrumentos para medirlos (Bartholomew et al. (2011)). La contraparte de las variables latentes son las variables manifiestas que sí son observables y directamente medibles, como pueden ser el número de respuestas correctas de un individuo en una evaluación académica de opción múltiple o el tiempo que se demora en resolver una tarea.

2.3. Modelos de Diagnóstico Cognitivo (MDC)

Los Modelos de Diagnóstico Cognitivo (MDC) son modelos de clasificación de variables latentes que relacionan las respuestas del individuo a un instrumento de evaluación en un cuestionario, con determinadas variables categóricas latentes que suelen ser binarias o dicotómicas (Templin y Henson (2006)). Esto permite agrupar a los evaluados en categorías no directamente observables (latentes) a las que tengan mayor probabilidad de pertenencia. Esto es muy valioso para identificar y explicar un fenómeno que a primera vista no tiene una explicación manifiesta. Algunos de los MDC más usados son: el DINA (Junker y Sijtsma (2001)), DINO (Templin y Henson (2006)) y el LLM (Davier (2024)).

Empecemos detallando dos de los MDC más sencillos y populares: el modelo DINA y el LLM. El modelo objetivo de esta tesis será formulado usando cada uno de estos; por consiguiente, presentaremos dos variaciones de dicho modelo. De forma general, será teóricamente posible adaptar otros MDC al modelo objetivo de acuerdo a los requerimientos de medición de la evaluación.

2.3.1. Modelo de Diagnóstico Cognitivo DINA

Sea Y_{in} la respuesta de un individuo examinado n ($n = 1,...,N$) al ítem i ($i=1,...,I$), q_{ik} un elemento dentro de una matriz Q de orden I x K; y $\boldsymbol{\alpha}_n = (\alpha_{n1}, \alpha_{n2}, ..., \alpha_{nk})$, un vector binario de k atributos latentes ($k = 1, ..., K$), donde *1*, en el k-ésima componente, significa que el individuo tienen dominio sobre el k-ésimo atributo y *0*, que no lo tiene. Este vector

latente es conocido también como el *estado de conocimiento del individuo n.*

El modelo requiere, como arriba introdujimos, de una matriz binaria de orden I x K, llamada matriz Q, donde su elemento q_{ik} de la fila i y de la columna k indica si la habilidad k es requerida para contestar al ítem i. Esta matriz sirve como una matriz de diseño cognitivo que especifica el nivel cognitivo que cada ítem debe tener para ser correctamente respondido (Torre (2009)). Esta matriz es usualmente definida exógenamente por un especialista en el área de evaluación.

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{bmatrix}$$

Por ejemplo, arriba, vemos una matriz Q de tres ítems y dos atributos latentes. Para que teóricamente un sujeto pueda responder correctamente el primer ítem, este debería dominar el primer atributo latente pero no el segundo. A diferencia del tercer ítem, donde se debe dominar el primero y el segundo a la vez.

El estado de conocimiento de un individuo n y la matriz Q producen una respuesta esperada η_{ni} para el ítem i igual a:

$$\eta_{ni} = \prod_{k=1}^{K} \alpha_{nk}^{q_{ik}} \tag{2.1}$$

De esta forma, la probabilidad que un individuo n, con su estado de conocimiento $\boldsymbol{\alpha}_n$, conteste correctamente al ítem i esta dado por:

$$P(\boldsymbol{\alpha}_n) = P(Y_{ni} = 1|\boldsymbol{\alpha}_n) = g_i^{1-\eta_{ni}}(1 - s_i)^{\eta_{ni}} \tag{2.2}$$

donde se introducen dos parámetros de ruido: un "parámetro de desliz" s_i, que denota a la probabilidad de que el individuo se equivoque al responder el ítem i a pesar de sí poseer todos los atributos requeridos; y un "parámetro de adivinanza" g_i, que denota la probabilidad de que el individuo conteste correctamente al ítem i así no tuviera el dominio de los atributos requeridos para el ítem.

El modelo DINA considera que para que un individuo responda correctamente a un ítem debe dominar todos los atributos requeridos por ese ítem (de acuerdo a lo indicado en la

matriz Q). Siguiendo con el ejemplo, el individuo solo estaría en capacidad de responder correctamente el tercer ítem si domina el ítem uno y dos, pero no tendría esta capacidad si conoce solo el ítem 1 o el ítem 2.

2.3.2. Modelo de Diagnóstico Cognitivo Logístico Lineal (LLM)

Otra clase de MDC, donde no hay necesidad de dominar el total de atributos requeridos por el ítem, y, por tanto, se cataloga como un modelo compensatorio, es el modelo Logístico Lineal o LLM (Maris (1999)). En este, cada atributo dominado en un ítem aumenta la probabilidad del individuo n de responder correctamente al ítem i. A diferencia del modelo DINA, donde el individuo debe dominar todos los atributos señalados por la matriz Q, en el LLM, el conocimiento parcial de los atributos requeridos por la matriz Q para un ítem no implica una probabilidad nula de éxito al tratar de responderlo.

Por ejemplo, evaluando el tercer ítem de la matriz Q mostrada antes (donde se deben dominar los atributos 1 y 2), teóricamente, un individuo podría responderlo correctamente si domina el primero, el segundo o ambos atributos a vez. El dominar solo uno de los dos genera alguna probabilidad de éxito en la respuesta, no una nula como en el caso del DINA, aunque no tanta como si se dominaran ambos a la vez.

El estado de conocimiento de un individuo n y la matriz Q producen una respuesta esperada $\zeta_n = \zeta_{ni}$ para el ítem i igual a:

$$\zeta_{ni} = \frac{\sum_{k=1}^{K} \alpha_{nk} \cdot q_{ik}}{\sum_{k=1}^{K} q_{ik}} \tag{2.3}$$

La probabilidad que un individuo n con su estado de conocimiento $\boldsymbol{\alpha}_n$ conteste correctamente al ítem i esta dado por:

$$P(\boldsymbol{\alpha}_n) = P(Y_{ni} = 1|\boldsymbol{\alpha}_n) = g_i^{1-\zeta_{ni}}(1 - s_i)^{\zeta_{ni}} \tag{2.4}$$

En la ecuación (2.3), apreciamos que la probabilidad de la respuesta esperada aumenta (tiende a 1) conforme más atributos requeridos por el ítem se dominan y viceversa. De esta manera, se explica por qué un individuo que domina más atributos requeridos tendrá mayor probabilidad de responderlo correctamente que otro que domina menos. Del mismo modo,

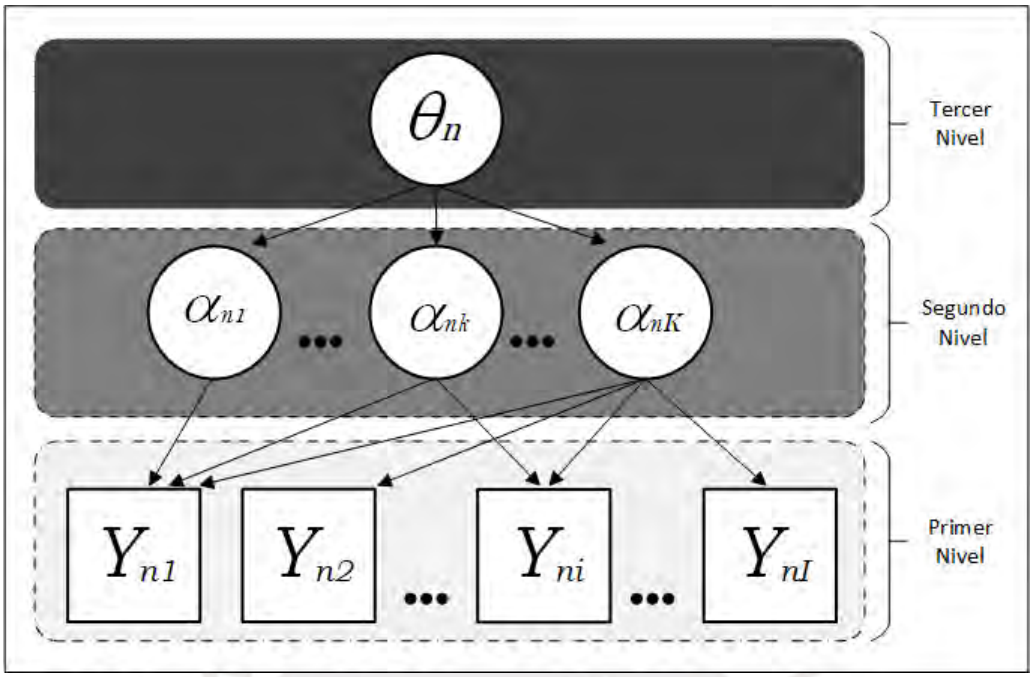
este individuo tendrá una menor probabilidad de éxito que alguien que domina todos los atributos requeridos.

2.4. Modelos de Diagnóstico Cognitivo Jerárquicos de Orden Superior

Los MDC Jerárquicos de Orden Superior (MDCJOS) fueron postulados para establecer una relación conjunta entre la habilidad general y el conocimiento específico sobre ciertos atributos de los individuos evaluados (Torre y Douglas (2004)). En este tipo de modelos, se calcula la probabilidad conjunta de responder correctamente los ítems evaluados dada la existencia de tres niveles cognitivos, desde la habilidad más general (tercer nivel) a la más específica (primer nivel). En la Figura 2.1, se aprecian la jerarquía existente entre los niveles para un individuo.

Comenzando de las más general a la más específica, el nivel de tercer nivel modela mediante θ la habilidad latente general de un individuo, la cual es una habilidad siempre presente en este cuando realiza alguna actividad. Luego, en el segundo nivel se describen la probabilidades de éxito m_k de dominar cada atributo k (k=1,...,K) dada la habilidad general. Esta permite determinar el estado de conocimiento $\boldsymbol{\alpha}_n = (\alpha_{n1}$, ... ,α_{nk}, ... ,$\alpha_{nK})$ del individuo n. Finalmente, en el primer nivel se calculan las probabilidades de acertar o errar un ítem i (i=1,...,I) dadas la habilidad general θ y el conocimiento de los atributos sobre cada ítem. De este modo, obtenemos el conjunto de respuestas $\boldsymbol{Y}_n = (Y_{n1}$, ... , Y_{ni}, ... ,$Y_{nI})$ dado los niveles superiores.

Figura 2.1: Esquema de MDC de Orden Superior para el individuo n en un periodo t



Adaptado de Zhan et al. (2019)

Vale destacar que, en el primer nivel, se puede optar por usar diferentes componentes de medición. Este hace referencia al modelo estadístico empleado para analizar los patrones de respuesta que unen las habilidades latentes del individuo n a las variables observadas (respuestas de la evaluación Y_n). Por ejemplo, el uso del modelo DINA como componente de medición (el cual es no compensatorio) permite calcular las probabilidades de responder correctamente los ítems considerando la habilidad general del individuo y su dominio completo de los atributos requeridos en cada ítem.

El modelo del componente de medición puede ser cambiado para flexibilizar al MDC. Como comentamos anteriormente, existen dos tipos de modelos generalmente usados para la medición: "compensatorios" y "no compensatorios" (o también llamados conjuntivos). El primero asume que el dominio de unos atributos latentes requeridos por un ítem compensa al no dominio de los otros; mientras que el segundo, presupone la necesidad del dominio de todos los atributos latentes para responder correctamente al mismo ítem.

Por ejemplo, si un ítem evaluado requiere del dominio de dos atributos (A1 y A2) para poder ser resuelto correctamente, este generaría tres escenarios posibles: que el individuo conozca ambos (A1 y A2), que solo conozca uno de los dos (domine A1 y no A2, o viceversa), o que no conozca ninguno de los dos. En el primer escenario, el individuo cumple todos

los requisitos y, por tanto, debería poder teóricamente resolver el ítem sin problemas. En el otro extremo, en el tercer escenario, este no debería poder teóricamente resolverlo porque no domina ninguno. Sin embargo, en el segundo caso, existen dos formas de calcular la probabilidad dependiendo del modelo usado en el componente de medición.

Por un lado, al usar el modelo compensatorio, se asume que el individuo obtiene una probabilidad parcial de resolver correctamente el ítem dado que conoce uno de los dos atributos necesarios; por lo que, se dice que el atributo dominado podría compensar al no dominado. Por otro lado, de usarse el modelo no compensatorio, se asume que el individuo no podrá teóricamente resolver el ítem correctamente si no cuenta con ambos atributos.

A pesar de no dominar los atributos requeridos, siempre se considera que el individuo posee una probabilidad reducida de éxito, expresada por el parámetros de "adivinanza" g (que conteste bien sin tener las competencias necesarias para hacerlo). Igualmente, se le calcula una probabilidad de errar, representada por el parámetros de "desliz" s (responder erróneamente, aún cuando se domine todos o algunos de los atributos necesarios).

Para entender mejor este modelo, imaginemos que queremos calcular la probabilidad de responder correctamente a una serie de preguntas sobre operaciones combinadas con números racionales a través de un examen. Para hacerlo, podemos recurrir a este modelo, considerando cada uno de sus niveles. El tercer nivel (orden superior) describe la habilidad general del individuo para responder preguntas (ítems) matemáticas. Esta capacidad influenciará al segundo nivel, el cual representa la capacidad de dominio sobre los atributos evaluados, como el conocimiento de las propiedades de los números reales, adición y sustracción, multiplicación y división; y operaciones combinadas de números reales. Asimismo, el dominio del individuo sobre estos atributos impactará al primer nivel (orden inferior), el cuál refleja su capacidad de desempeño durante la resolución del examen. Note que la introducción de parámetros de "ruido" implica que el individuo podría equivocarse a pesar de saber sobre el tema o, de modo inverso, podría resolver los ítems adivinándolos, sin conocerlos.

De esta manera, el MDCJOS recoge la influencia jerárquica de sus tres niveles: la habilidad general (tercer nivel) influye en el dominio de los atributos (segundo nivel) y, a su vez, estas influyen en la habilidad del individuo para resolver cada uno de los ítems evaluados (primer nivel).

2.5. Modelos de Diagnóstico Cognitivo Longitudinales

Las evaluaciones longitudinales son aquellas que se realizan durante distintos puntos periodos para identificar los cambios del estado de conocimiento de los individuos. Debido a la necesidad de poder tener un instrumento que se adapte a estas evaluaciones múltiples en el tiempo, se desarrollaron variaciones a los MDC clásicos. Así, se postularon los modelos diagnóstico cognitivo longitudinal (MDCL), como el sLong-DINA (presentado por Zhan et al. (2019)) y el sLong-LLM (presentado por Zhan y He (2021)). Estos están diseñados para estimar la probabilidad que un individuo domine ciertos atributos latentes a través de distintos periodos en el que el individuo es evaluado (Zhan et al. (2019)).

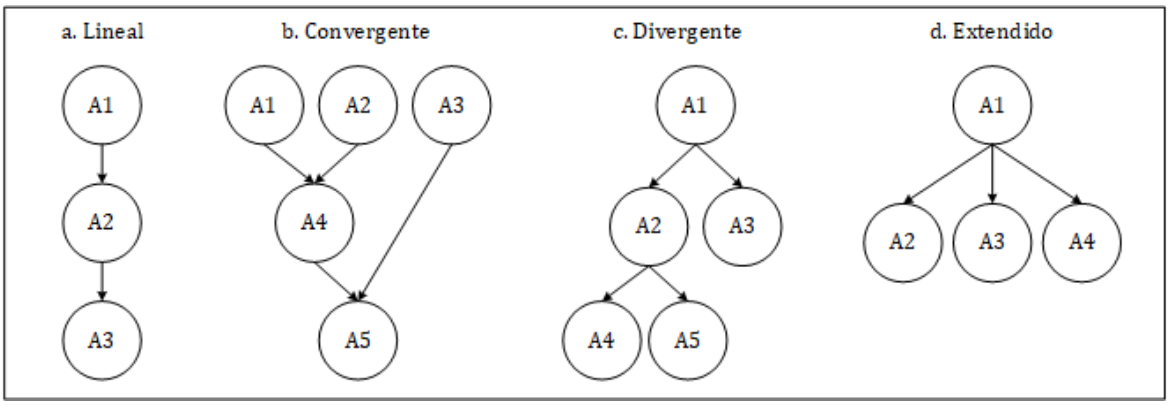
El sLong-DINA y el sLong-LLM son extensiones longitudinales de los modelos de Orden Superior DINA (presentado por Torre y Douglas (2004)) y LLM, respectivamente. Por lo que, poseen dos cualidades adicionales a los modelos clásicos DINA y LLM: pueden realizar evaluaciones longitudinales y cuentan con los tres niveles jerárquicos que vimos en el punto anterior. En el siguiente capítulo, mostraremos su formulación matemática.

2.6. Modelo Jerárquico Secuencial

Así como existe una jerarquía de niveles, también hay otra jerarquía entre los atributos evaluados que robustece al MDCJOS. El MDC Jerárquico de Orden Superior Secuencial introduce dependencias entre los atributos de segundo nivel; pues los atributos pueden naturalmente tener dependencias entre sí. De modo que, el dominar uno de estos incrementa la probabilidad de dominar otro que es dependiente del primero.

En la Figura 2.2, se muestran cuatro estructuras jerárquicas de seis atributos latentes en las que se ilustra las relaciones secuenciales entre atributos. En el primer caso (*Lineal*), cada atributo es dependiente del anterior. En el segundo caso (*Convergente*), se empieza con una secuencia lineal (A1 $\rightarrow$ A2, es decir, A2 depende de A1); luego, aparecen dos atributos dependientes del anterior (A3 y A4 los que a su vez tienen un efecto sobre el dominio del siguiente (A5); finalmente, ocurre una dependencia de A6 con A5. En el tercer caso (*Divergente*), cada atributo depende del nivel anterior: A3 depende de A2; A5 y A6 dependen de A4; a su vez, A2 y A4 dependen de A1. En el cuarto caso (*Extendido*), todos se ven afectados por el primer atributo A1.

Figura 2.2: Modelos de Jerarquías de Atributos



Adaptado de: Leighton et al. (2004).

La jerarquía de atributos son asunciones teóricas que permiten identificar cuál combinación de atributos, o *perfil de atributos*[2] ("attribute profile" en inglés), en adelante "perfil", es posible y cuál no lo es. Normalmente, para que un atributo sea dominado, el atributo que lo precede (del que depende) debe ser dominado primero. Por lo que, cualquier combinación que no cumpla con reflejar las dependencias entre atributos será imposible que exista. De esta manera, se generan dos escenarios: *perfil de atributos permisibles* y *perfil de atributos imposibles*. Los primeros son perfiles que pueden existir dadas las dependencias entre los atributos; los segundos son aquellos que no pueden suceder porque no respetan dichas dependencias.

Por ejemplo, revisando la estructura secuencial lineal de la Figura 2.2 para 3 atributos, los perfiles permisibles considerarían que el atributo A1 no requiere de ningún otro atributo para ser dominado, solo de sí mismo. Mientras que el A2, requiere del A1; y el A3, del A2 (que a su vez depende del A1). Por el otro lado, los perfiles imposibles serán aquellos que consideran al dominio del atributo A3 sin dominar previamente el A2 y del dominio de A2 sin el A1.

Los perfiles se denotarán mediante una secuencia de k dígitos, que pueden ser "1" (si hay dependencia) o "0" (si no la hay), donde la primera columna hace referencia al primer atributo, la segunda columna al segundo atributo y así en adelante. Siguiendo el ejemplo, como A1 no depende de otro atributo, su perfil se escribe: "100". El "1" en el primer elemento de la secuencia indica que depende de sí mismo (solo A1); los "0" siguientes, que no depende del segundo (A2) ni del tercer atributo (A3). Análogamente, el perfil de A2 será "110" y el de A3, "111". Asimismo, se debe considerar el caso donde no se requiere dominar ningún

[2]Se puede calcular la cantidad de perfiles totales mediante 2^k, donde k es igual a la cantidad de atributos evaluados.

atributo: “000”. De esta manera, con 3 atributo solo existen 4 perfiles permisibles, denotado como C^*=4; el resto de perfiles ($2^3 - 4 = 4$: “001”, “010”, “011”, “101”) son imposibles pues no respetan las dependencias de la jerarquía lineal entre los atributos.

Utilizando la notación anterior, se puede ver claramente las diferencias del uso de una estructura jerárquica secuencial en el modelamiento de los MDL. En el ejemplo dado, se generan solo 4 perfiles permisibles, denotado como C^* (en este caso, $C^* = 4$), donde C es la cantidad de perfiles totales (en este caso, $C = 2^3 = 8$). Si no se usara esta estructura jerárquica, existirían 8 perfiles; lo que implicaría el cálculo de 4 perfiles imposibles.

Repasando lo expuesto, vemos que tanto los modelos estructurales secuenciales como los longitudinales de orden superior se fundamentan en estructuras latentes jerárquicas (Zhan et al. (2020)). Al compartir esta base conceptual, la integración de ambos modelos nos permite realizar un cálculo más preciso de la probabilidad de dominio del individuo sobre los atributos evaluados a través de las diferentes órdenes de conocimiento.

Hasta hace poco, los MDCL no incluían en su modelamiento estos conceptos. Este fue propuesto inicialmente por Zhan et al. (2019) y luego ampliado por Zhan y He (2021), con un modelamiento longitudinal, que consideraba los parámetros de evaluaciones en diferentes puntos de tiempo, expandiendo los modelos bases DINA y Logístico Lineal (LLM).

2.7. Modelos aplicados

Para aplicar los conceptos explicados sobre los modelos de diagnóstico cognitivo longitudinal con estructura jerárquica de orden superior y atributos dependientes, se integraron diferentes modelos desarrollados. Primero, se consideró el modelo Orden Superior DINA (o “Higher Order DINA”, presentado por Torre y Douglas (2004)) que aportó la estructura jerárquica de órdenes superiores. Luego, Zhan et al. (2019) expandieron este modelo (llamado Longitudinal DINA) para que el individuo pueda ser evaluado en distintos momentos. Posteriormente, se desarrolló el “modelo de longitudinal de Orden Superior DINA”, llamado *Hi-Long-DINA*, que incorpora la estructura secuencial de atributos jerárquicos, en la que se considera las dependencias entre estos.

Del mismo modo, manteniendo el tercer y el segundo nivel, pero usando como componente de medición en el primer nivel al modelo Logístico Lineal (“LLM”), en vez del DINA, y

considerando las mediciones longitudinales, formaron el "modelo de longitudinal de Orden Superior Logístico Lineal", llamado *Hi-Long-LLM*. Este difiere del Hi-Long-DINA en que es un modelo compensatorio, es decir, el individuo tiene capacidad de resolver ítems, a pesar de, no dominar todos los atributos requeridos (los atributos dominados compensan a los no dominados).

En el siguiente capítulo, repasaremos con mayor profundidad la teoría del modelo que integra a todos los vistos en este capítulo.

Capítulo 3

Modelos de diagnóstico cognitivo longitudinal de orden superior secuenciales

En años recientes, los modelos cognitivos han ganado atención por su capacidad para medir el estado actual de conocimiento de los estudiantes. A pesar que estos buscan promover la capacidad de aprendizaje de los estudiantes en base a la retroalimentación de los diagnósticos realizados, son pocas investigaciones las que han logrado evaluar si este realmente ha generado los resultados deseados. Ello debido a que la forma de recolección de datos de los individuos evaluados se toma una sola vez en el tiempo (estudio "cross-sectional") (Zhan y He (2021)). De modo contrario, el "Modelo de Diagnóstico Cognitivo Longitudinal" (o MDCL) evalúa a los mismos individuos durante un periodo de tiempo para poder evaluar sus cambios.

De este modo, los MDC pueden usar estudios "cross-sectional" o longitudinales. Las principales diferencias entre ambos tipos de estudios son la facilidad de la toma de muestra y la evaluación del aprendizaje. En el primer caso, es más sencillo recoger datos de la muestra pues solo se realiza una vez, mientras que, en el segundo caso, varias veces al mismo individuo. Por otro lado, la evaluación del aprendizaje en el primer caso, no permite medir el crecimiento individual en el aprendizaje; a diferencia del segundo, donde sí se identifican sus fortalezas y debilidades dentro del periodo de tiempo que se realizó el estudio (Zhan y He (2021)).

Como vimos en el capítulo anterior, para asegurar que se consideren longitudinalmente los órdenes superiores (tercer nivel, que recoge al conocimiento general del individuo, y se-

gundo nivel, que recoge su dominio en los atributos evaluados) y la jerarquía secuencial de órdenes (dependencias entre atributos), Zhan y He (2021) postulan el *Modelo de Diagnóstico Cognitivo Longitudinal de Orden Superior Secuencial* (MDCLOSS o, en inglés, "Sequential Higher Order Longitudinal Cognitive Diagnostic Model"). De esta manera, el MDCLOSS es complementado con el modelo de estructura secuencial que permite calcular las probabilidades de dominio de los atributos evaluados, dada las relaciones de dependencia existente entre estos.

3.1. Formulación del Modelo

En la Figura 2.1, se ilustraron los tres niveles que se usan en los MDCLOSS, del más general al más específico: el tercer nivel modela la habilidad latente general de un individuo n; el segundo nivel modela la habilidad del individuo para dominar los K atributos evaluados dada su habilidad latente general; y el primer nivel, la capacidad específica del individuo para responder correctamente a un ítem i dado su dominio sobre los atributos requeridos. Es importante comentar que, además de considerar la jerarquía de los niveles cognitivos, el MDCLOSS permite también la evaluación longitudinal.

Los niveles en el MDCLOSS se expresan de la siguiente forma:

3.1.1. Tercer nivel

En este nivel de orden superior, la habilidad general de un individuo $n \in \{1, 2, ..., N\}$ se modela mediante el vector aleatorio latente:

$$\boldsymbol{\theta}_n = (\theta_{n1}, ..., \theta_{nT})' \sim MVN_T(\boldsymbol{\mu}_{\boldsymbol{\theta}}, \boldsymbol{\Sigma}_{\boldsymbol{\theta}}), \tag{3.1}$$

donde θ_{nt} corresponde a la habilidad latente general de este individuo n en el periodo $t \in \{1, 2, ..., T\}$. Así, la habilidad general es modelada en T periodos mediante una distribución normal multivariada con un vector de medias $\boldsymbol{\mu} = (\mu_1, ..., \mu_T)'$ y matriz de varianza-covarianza

$$\boldsymbol{\Sigma} = \begin{bmatrix} \sigma_1^2 & \sigma_{12}... & \cdots & \sigma_{1T} \\ \sigma_{21} & \sigma_2^2 & \cdots & \sigma_{2T} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{T1} & \sigma_{T2} & \cdots & \sigma_T^2 \end{bmatrix},$$

donde σ_{st} es la covarianza de la habilidad general en los periodos s y t, y σ_t^2 es la varianza de esta habilidad en el periodo t. Note que, marginalmente, $\theta_{nt} \sim N(\mu_t, \sigma_t^2)$.

3.1.2. Segundo nivel

Para cada atributo latente $k \in \{1, 2, ..., K\}$, se introducen, en este nivel, los parámetros ξ_k y β_k. De tal manera que, la probabilidad de que un individuo n domine el atributo k en el periodo t, dada su habilidad general, viene dada por:

$$m_{nkt} = P(\alpha_{nkt} = 1|\theta_{nt}; \xi_k, \beta_{it}) = \frac{exp(\xi_k\theta_{nt} - \beta_k)}{1 + exp(\xi_k\theta_{nt} - \beta_k)}, \tag{3.2}$$

representando $\alpha_{nkt} \in \{0, 1\}$ el dominio del atributo k del individuo n en el punto t. Aquí, θ_{nt} es la habilidad latente general del individuo n en el periodo t, ξ_k es un parámetro de discriminación del atributo k con relación a la habilidad general y β_k es un parámetro de dificultad del atributo k para todos los periodos (pues el modelo asume una invarianza en la estructura latente a lo largo del tiempo). En este nivel, consideramos que los α_{nkt} son independientes entre sí para los distintos atributos dado que la habilidad general θ_{nt} se mantiene constante para el mismo individuo en el mismo periodo.

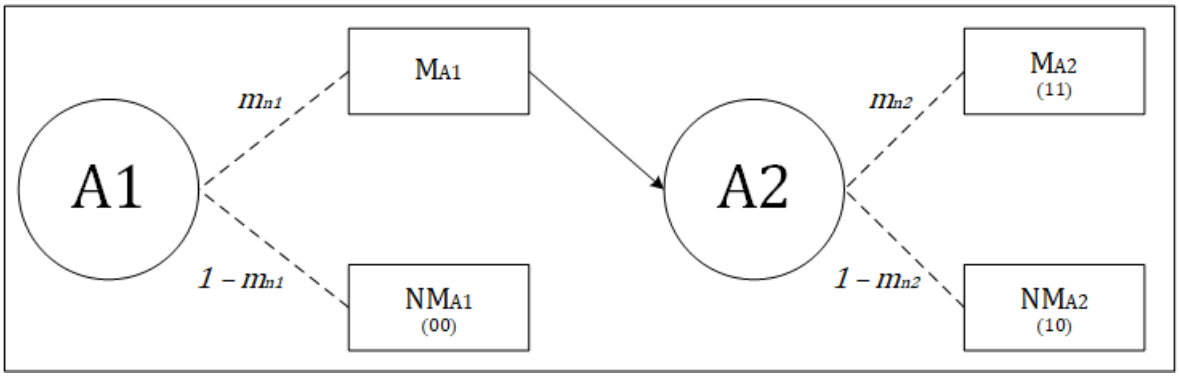
Anteriormente, para cada periodo de tiempo, los MDC convencionales (como los modelos de rasgos latentes de Orden Superior, propuestos por Torre y Douglas (2004), y adaptados a MDCL por Zhan et al. (2019)) permitían la existencia de $C = 2^K$ perfiles de atributos posibles. Allí, la probabilidad de que el individuo n tenga un perfil de atributos $\boldsymbol{c}$ ($c \in \{1, 2, ..., C\}$) en el periodo t era calculado por:

$$\pi_{n\boldsymbol{c}t} = P(\boldsymbol{\alpha}_{nt} = \boldsymbol{c}) = \prod_{k=1}^{K} m_{nkt}^{\alpha_{ckt}}(1 - m_{nkt})^{1-\alpha_{ckt}}, \tag{3.3}$$

donde $\pi_{n\boldsymbol{c}t}$, también conocida como probabilidad combinada; es la probabilidad que un individuo n en un tiempo t tenga un perfil de atributos $\boldsymbol{c}$. Esta probabilidad satisface que $\sum_{\boldsymbol{c}=1}^{C} \pi_{n\boldsymbol{c}t} = 1$. También, $\alpha_{nkt} \in \{0, 1\}$ dependiendo si el perfil $\boldsymbol{c}$ denota no dominio o dominio del atributo k, respectivamente.

En la Figura 3.1, se esquematiza, a modo de diagrama de árbol, la forma de modelar la probabilidad π_{nct} (ecuación 3.3) en una estructura jerárquica de atributos lineal, donde

Figura 3.1: Proceso de dominio secuencial lineal de dos atributos



Adaptado de: Zhan et al. (2020).

cada nivel contiene: atributos (círculos), procesos de dominio (ramas) y resultados de proceso (rectángulos). Los niveles posteriores dependen de los anteriores para existir. Esta relación entre niveles donde es necesario el conocimiento de los atributos padres para dominar a los atributos hijos es referida como "dominio secuencial" (de orden superior).

En la Figura 3.1, usamos dos atributos (A1 y A2), donde el dominio de A2 depende de A1 (indicado con la flecha sólida que va hacia A2). Los atributos pueden generar dos estados que se muestran en los rectángulos [3]: dominio (MA) o no dominio (NMA) del atributo. Asimismo, las líneas punteadas representan la probabilidad del individuo n de dominar o no el atributo k, donde m_{nk} es la probabilidad dominio.

Siguiendo la lógica secuencial del ejemplo, el cálculo de las probabilidades se realiza mediante la multiplicación de los perfiles de atributos permisibles. Es decir, solo cuando el atributo padre es dominado, se puede considerar las probabilidades del atributo hijo. En el ejemplo, la probabilidad de dominar A1 (el primer atributo) es m_{n1}, y de no dominarlo, 1–m_{n1}. Si A1 es dominado, entonces se se calcula las probabilidades de A2 considerando la probabilidad exitosa de dominio de A1.

Viendo la Figura 3.1, observamos que los perfiles permisibles $\boldsymbol{c}$ en el periodo t son [4]: "00", "10" y "11"; y sus respectivas probabilidades son: $\pi_{n(00)}$, $\pi_{n(10)}$ y $\pi_{n(11)}$. De esta manera, $\pi_{n(00)} = (1 - m_{n1})$, $\pi_{n(10)} = (m_{n1}) \cdot (1 - m_{n1})$, y $\pi_{n(11)} = (m_{n1}) \cdot (m_{n2})$. La probabilidad combinada suma 1: $\pi_{n(00)} + \pi_{n(10)} + \pi_{n(11)} = 1$. Del mismo modo, se pueden diseñar diferentes estructuras secuenciales para otras jerarquías de atributos, como la convergente, divergente o extendida

[3] Dentro de cada rectángulo, junto con el estado de dominio (MA) o no dominio (NMA), se agrega el perfil de atributos que lo genera: "00" = no dominio del A1 ni A2; "10" = dominio de A1 pero no de A2; "11" = dominio de A1 y A2.

[4] El patrón de atributos "01" es un perfil imposible porque dominar A1 es necesario para dominar A2.

que fueron presentadas en la Figura 2.2.

La ecuación (3.3) considera a todas las combinaciones de perfiles que podrían existir, aun cuando hay algunas que podrían no existir. En otras palabras, no se respeta las dependencias entre sus atributos. Debido a esta desventaja, Zhan et al. (2020) propusieron una nueva forma de calcular esta probabilidad combinada:

$$\pi_{n\boldsymbol{c}^*t} = \prod_{k=1}^{K} \left[m_{nkt}^{Z_{\boldsymbol{c}^*kt}} \cdot (1 - m_{nkt})^{1 - Z_{\boldsymbol{c}^*kt}} \right]^{d_{\boldsymbol{c}^*kt}}, \tag{3.4}$$

donde $Z_{\boldsymbol{c}^*kt} \in \{0, 1\}$ es el elemento de una matriz Z_t que describe el no dominio o dominio, respectivamente, del atributo k en el perfil permisible $\boldsymbol{c}^*$ para el tiempo t. La matriz Z_t, de dimensión C^* x K, registra asociación entre los perfiles de atributos permisibles y el resultado del proceso de dominio de los atributos. La Figura 3.2 presenta como ejemplo la estructura jerárquica lineal presentada en la Figura 3.1. Aquí $Z_{\boldsymbol{c}^*kt} = 1$ indica que en el perfil $\boldsymbol{c}^*$ se domina el atributo k; mientras que $Z_{\boldsymbol{c}^*k} = 0$ indica que no se lo domina. Si es que $Z_{\boldsymbol{c}^*kt} = NA$ significa que el perfil $\boldsymbol{c}^*$ no admite el dominio del atributo k porque no cumple con los requerimientos. En el ejemplo, para un tiempo t, vemos que el atributo 2 (A2), en la primera fila, no puede ser dominado sin el dominio previo del A1. En la segunda y tercera fila, se describen los escenarios restantes: se domina el A1 y no el A2, o se domina el A1 y luego el A2.

Figura 3.2: Ejemplo de Matriz Z y Matriz D para estructura jerárquica lineal de la Figura 3.1

$$Z_t = \begin{bmatrix} 0 & NA \\ 1 & 0 \\ 1 & 1 \end{bmatrix}, \quad D_t = \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 1 & 1 \end{bmatrix}$$

Tomado de: Elaboración propia.

Asimismo, $d_{\boldsymbol{c}^*kt}$ es el elemento de la matriz D_t que especifica las combinaciones de perfiles y atributos permisibles, indicando $d_{\boldsymbol{c}^*kt} = 1$ que el atributo k tiene sentido en el perfil $\boldsymbol{c}^*$. La matriz D_t es también de dimensión C^* x K. Note que $Z_{\boldsymbol{c}^*kt} = NA$ indica que un perfil no es permisible, entonces $d_{\boldsymbol{c}^*kt} = 0$.

3.1.3. Primer nivel

Como se mencionó en el capítulo anterior, el primer nivel se expresa de acuerdo al componente de medición usado. Zhan y He (2021) presentaron su modelo MDCLOSS con dos aplicaciones: el Modelo Longitudinal Jerárquico DINA (Hi-Long-DINA), que es la expansión del modelo Longitudinal DINA, y el modelo Longitudinal Jerárquico Lineal Logístico (Hi-Long-LLM), proveniente del modelo longitudinal Logístico Lineal. Mientras que el primero es un modelo conjuntivo que presupone el dominio de todos los atributos latentes para responder correctamente un ítem; el segundo es un modelo compensatorio, en el que se asume que la maestría de unos atributos latentes compensa al no dominio de los otros. Asimismo, dicho componente de medición puede ser cambiado en la ecuación de verosimilitud por otro MDC.

Modelo Hi-Long-DINA

Sea Y_{nit} la respuesta del individuo n (n=1, ..., N) al ítem i (i=1, ..., I) en el periodo t (t=1, ..., T), donde $Y_{nit} \in \{0,1\}$ (1 si tuvo una respuesta correcta o 0 si fue errada). El modelo Jerárquico Longitudinal DINA, abreviado como Hi-Long-DINA, puede ser expresado a través de las tres ecuaciones (3.1), (3.2) y (3.5) según su nivel jerárquico que describiremos a continuación; pero en donde su inferencia se realizará con las restricciones descritas en (3.4).

El primer nivel para el modelo **Hi-Long-DINA** se expresa por:

$$p_{nit} = P(Y_{nit} = 1|\boldsymbol{\alpha}_{nt}; g_{it}, s_{it}) = \frac{exp(\lambda_{0it} + \lambda_{1it} \prod_{k=1}^{K} \alpha_{nkt}^{q_{ikt}})}{1 + exp(\lambda_{0it} + \lambda_{1it} \prod_{k=1}^{K} \alpha_{nkt}^{q_{ikt}})}, \tag{3.5}$$

donde p_{nit} es la probabilidad que el individuo n haya respondido correctamente al ítem i en el periodo t; $\boldsymbol{\alpha}_{nt}$=(α_{n1t},...,α_{nKt})$'$ es el estado de conocimiento del individuo n en el periodo t (con $\alpha_{nkt} \in \{0,1\}$); q_{ikt} es la entrada (i,k) de la matriz Q en el periodo t e indica el requerimiento del atributo k para el ítem i en el tiempo t, $q_{ikt} \in \{0,1\}$; λ_{0it} es un parámetro de intercepto y λ_{1it} es un parámetro de interacción para el ítem i en un tiempo t.

Para efectos prácticos, Zhan y He (2021) reparametrizó a g_{it} y s_{it}, en términos de λ_{0it} y λ_{1it} mediante las ecuaciones (3.6) a (3.8) y (3.9) a (3.12), respectivamente. El parámetro λ_{0it} representa el log odds de adivinar la respuesta correcta en un ítem, sin dominar los atributos necesarios; mientras que λ_{1it} muestra el log odds de responder erróneamente, dominando los atributos requeridos.

$$g_{it} = P(Y_{nit} = 1|\boldsymbol{\alpha}_{nt} = 0) \tag{3.6}$$

$$\lambda_{0it} = logit(g_{it}) = log\left(\frac{g_{it}}{1 - g_{it}}\right) \tag{3.7}$$

$$g_{it} = \frac{e^{\lambda_{0it}}}{1 + e^{\lambda_{0it}}} \tag{3.8}$$

$$s_{it} = P(Y_{nit} = 0|\boldsymbol{\alpha}_{nt} = 1) \tag{3.9}$$

$$\lambda_{1it} = logit(1 - s_{it}) = g_{it} + s_{it} \tag{3.10}$$

$$log\left(\frac{s_{it}}{1 - s_{it}}\right) = g_{it} + s_{it} \tag{3.11}$$

$$s_{it} = 1 - \frac{e^{\lambda_{0it}+\lambda_{1it}}}{1 + e^{\lambda_{0it}+\lambda_{1it}}} \tag{3.12}$$

Debido al uso de la función de enlace logístico en este modelo, mediante una reparametrización, ver ecuaciones de (3.6) a (3.8) y de (3.9) a (3.12), se redefinieron los *parámetros de adivinanza* y de *desliz* del ítem i en el punto t para este modelo como: $g_{it} = \frac{e^{\lambda_{0it}}}{1+e^{\lambda_{0it}}}$ y $s_{it} = 1 - \frac{e^{(\lambda_{0it}+\lambda_{1it})}}{1+e^{(\lambda_{0it}+\lambda_{1it})}}$, respectivamente. El parámetro de adivinanza representa la probabilidad que el individuo conteste correctamente un ítem sin dominar sus atributos requeridos. El parámetro de desliz indica la probabilidad que el individuo se equivoque al responder un ítem a pesar de sí dominar sus atributos requeridos.

Así como en este modelo el primer nivel utiliza el modelo DINA; igualmente, podría ser usado otro MDC que se crea conveniente. Podría ser un modelo cercano, como el GDINA o DINO, o uno diferente LLM, que veremos a continuación.

Modelo Hi-Long-LLM

Manteniendo las ecuaciones de tercer y segundo nivel, se puede utilizar el modelo Logístico Lineal, abreviado como Hi-Long-LLM, en el primer nivel como componente de medición. A diferencia del Hi-Long-DINA donde se exige que el individuo domine todos los atributos requeridos por el ítem; en este, se consideran los atributos dominados por el individuo que requiere el ítem, a pesar que este no tenga todos los atributos necesarios. De esta manera, el individuo evaluado consigue un puntaje compensatorio por su dominio parcial del ítem, reflejado en los atributos latentes que conoce. El Hi-Long-LLM es expresado a través de las

tres ecuaciones (3.1), (3.2) y (3.6) según su nivel jerárquico; pero en donde su inferencia se realizará con las restricciones descritas en (3.4).

El primer nivel para el modelo **Hi-Long-LLM** se expresa como:

$$p_{nit} = P(Y_{nit} = 1|\boldsymbol{\alpha}_{nt}; \lambda_{0it}, \lambda_{1ikt}) = \frac{exp(\lambda_{0it} + \sum_{k=1}^{K} \lambda_{1ikt}.\alpha_{nkt}.q_{ikt})}{1 + exp(\lambda_{0it} + \sum_{k=1}^{K} \lambda_{1ikt}.\alpha_{nkt}.q_{ikt})}, \tag{3.13}$$

donde λ_{0it} es un parámetro del intercepto; λ_{1ikt} es un parámetro de efecto principal; y α_{nkt} y q_{ikt} mantienen la misma interpretación que en el modelo anterior. Al considerarse una sumatoria $\sum_{k=1}^{K} \lambda_{1ikt}.\alpha_{nkt}.q_{ikt}$, el no dominar un atributo k requerido por un ítem no se castiga con una probabilidad nula, sino que se considera la suma de las probabilidades de los atributos que sí se domina. Asimismo, en el tiempo t, la probabilidad de responder correctamente aumenta en función del dominio del individuo de cada atributo requerido y del efecto generado por el parámetro λ_{1ikt}.

De las ecuaciones (3.6) a (3.8) y (3.9) a (3.12), se puede deducir g_{it} y s_{it} respectivamente para el modelo Hi-Long-LLM en términos de λ_{0it} y λ_{1ikt}. De esta manera, obtenemos: $g_{it} = \frac{exp(\lambda_{0it})}{1+exp(\lambda_{0it})}$, y $s_{it} = 1 - \frac{exp(\lambda_{0it}+\sum_{k=1}^{K} \lambda_{1ikt}.\alpha_{nkt}.q_{ikt})}{1+exp(\lambda_{0it}+\sum_{k=1}^{K} \lambda_{1ikt}.\alpha_{nkt}.q_{ikt})}$ son los parámetros de adivinanza y desliz, respectivamente, para el ítem i en el periodo t.

Modelos sLong-DINA y sLong-LLM

Otros modelos similares al Hi-Long-DINA y Hi-Long-LLM comentados brevemente en el capítulo 2 fueron el sLong-DINA y el sLong-LLM. Estos son definidos usando las ecuaciones que vimos: (3.1), (3.2), (3.5) y (3.1), (3.2), (3.13), respectivamente. La diferencia principal entre ambos pares de modelos radica en que los segundos no cuentan con ninguna restricción para solo considerar los perfiles permisibles (C^*) de acuerdo a una jerarquía secuencial definida, sino que emplean todos los perfiles posibles ($C = 2^K$) sin depender de esta.

3.2. Inferencia del Modelo

Considerando el concepto de independencia local, la función de **verosimilitud** en una muestra observada de respuestas bajo el MDCLOSS, puede expresarse como:

$$\mathcal{L}(\pi_t, \Omega_t) = \prod_{n=1}^{N} \sum_{\boldsymbol{c}^*=1}^{C^*} \pi_{n\boldsymbol{c}^*t} \prod_{i=1}^{I_t} p_{nit}^{y_{nit}} (1 - p_{nit})^{1-y_{nit}} \tag{3.14}$$

donde las probabilidades p_{nit} en el modelo de medición vienen dadas por las ecuaciones (3.5) o (3.13); y_{nit} es la respuesta del individuo n al ítem i en el tiempo t; $\pi_{n\boldsymbol{c}^*t}$ es la probabilidad combinada de los perfiles de atributos permisibles $\boldsymbol{c}^*$ del individuo n en el periodo t, dada en (3.4); I_t es la cantidad de ítems i en el tiempo t; C^* es la cantidad de perfiles permisibles y N es la cantidad de individuos de la muestra. Asimismo, $\Omega_t = (\boldsymbol{g}_t, \boldsymbol{s}_t)'$ es un arreglo de parámetros de ítems, donde $\boldsymbol{g}_t$ y $\boldsymbol{s}_t$ son los vectores de parámetros de adivinanza y desliz respectivamente para los diferentes ítems en el tiempo t; $\boldsymbol{\pi}_t$ es el vector de probabilidades combinadas de los perfiles en el periodo t.

Para los casos del sLong-Dina y el sLong-LLM, que consideran todos los perfiles posibles, la función de verosimilitud usa C en vez de C^* y, por tanto, se expresa como:

$$\mathcal{L}(\pi_t, \Omega_t) = \prod_{n=1}^{N} \sum_{\boldsymbol{c}=1}^{C} \pi_{n\boldsymbol{c}t} \prod_{i=1}^{I_t} p_{nit}^{y_{nit}} (1 - p_{nit})^{1-y_{nit}} \tag{3.15}$$

Si bien hemos descrito la función de verosimilitud del modelo, su inferencia, dado la complejidad y gran cantidad de parámetros, es conveniente realizarla a través de métodos de cadenas de Markov de Monte Carlo. En este trabajo, se utilizará JAGS (Just Another Gibbs Sampler) en su implementación computacional. Los códigos utilizados pueden ser revisados en los anexos de este documento.

3.2.1. Inferencia bayesiana

La inferencia bayesiana es un método estadístico que actualiza las estimaciones de una hipótesis con parámetros desconocidos a medida que más información es generada y procesada. A diferencia de la estadística clásica, en la que los parámetros son fijos pero desconocidos, la inferencia bayesiana asigna probabilidades a los parámetros que permite incorporar conocimiento previo al análisis.

El concepto central en la inferencia bayesiana es el teorema de Bayes, el cual calcula la probabilidad a posteriori de un evento en base a los datos y a una probabilidad a priori dada al evento.

3.2.2. Cadenas de Markov Monte Carlo y Muestreo de Gibbs

La inferencia bayesiana busca calcular esencialmente la distribución a posteriori de los parámetros de cierta distribución. Normalmente, dicha distribución es muy compleja y difícil o imposible de evaluar. Para superar ello, los algoritmos de cadenas de Markov Monte Carlo (MCMC por sus sigas en inglés) proveen una solución computacionalmente viable. El MCMC es un método de muestreo computacional que permite generar valores de una distribución a posteriori sin necesidad de conocer sus propiedades matemáticas.

Las MCMC poseen dos componentes básicos: el método de Monte-Carlo y las Cadenas de Markov. Por un lado, el método de Monte-Carlo permite estimar propiedades de una distribución mediante la generación de muestras aleatorias de esta distribución. Su uso permite calcular estadísticos, como la media, de una muestra de gran tamaño con facilidad. Esto es muy beneficioso cuando es fácil tomar muestras de la distribución para encontrar alguna propiedad de ella, pero es complicado obtener analíticamente la distribución. Por otro lado, las cadenas de Markov postulan que las muestras están generadas por un proceso secuencial especial. Cada muestra aleatoria generada es usada como insumo para producir la siguiente muestra aleatoria, formando así una cadena. La propiedad especial de la cadena de Markov indica que, mientras que cada nueva muestra generada dependa de la anterior, las nuevas muestras no dependen de ninguna muestra anterior a la última (van Ravenzwaaij et al. (2016)).

El muestreo de Gibbs es un algoritmo particular MCMC, que es útil para trabajar problemas multidimensionales donde se requieren tomar muestras de distribuciones de probabilidad multivariada.

Este algoritmo divide en una cantidad determinada de subvectores al vector de parámetros del modelo (conformado por variables aleatorias) que se desea estimar. Iterativamente, genera muestras de uno de estos subvectores a partir de la distribución condicional completa de los otros subvectores. De esta manera, cada subvector se actualiza condicionado a los valores más recientes de los otros subvectores (Gelman et al. (2013)). Eventualmente, tras múltiples repeticiones, las muestras convergerán hacia la distribución conjunta deseada.

3.2.3. Just Another Gibbs Sampler (JAGS)

"Just Another Gibbs Sampler", abreviado como "JAGS", es un programa de software diseñado para realizar modelamiento e inferencia bayesiana y que sucedió al programa "BUGS" ("Bayesian inference Using Gibbs Samplings"). Principalmente, usa el algoritmo de muestreo de Gibbs, aunque podría utilizar otros dependiendo de la complejidad del modelo, como el Metropolis-Hastings.

JAGS construye muestras de MCMC para modelos jerárquicos complejos (Plummer (2003)). Este toma la verosimilitud del modelo y las distribuciones a priori de los parámetros y retorna una muestra de MCMC de la distribución *a posteriori* del modelo (Kruschke (2014)).

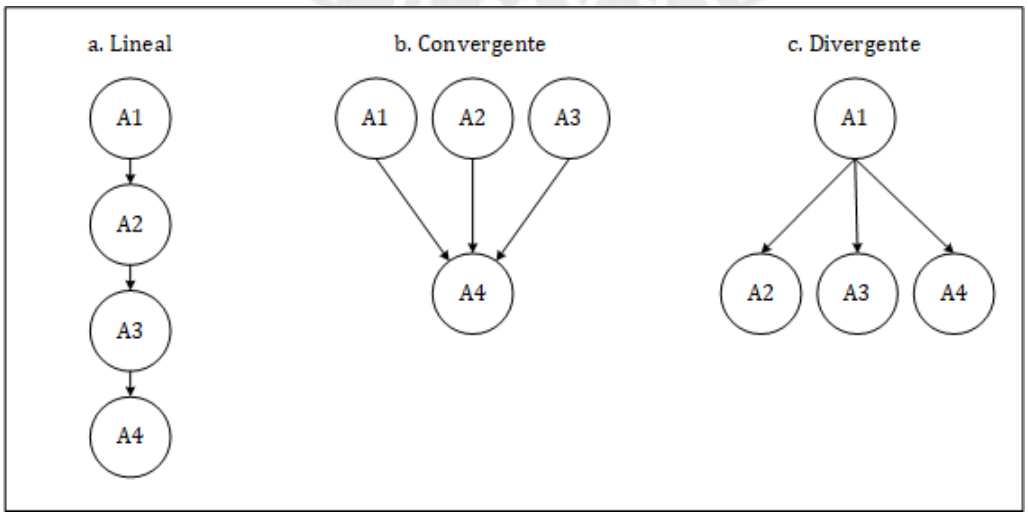
Capítulo 4

Estudio de Simulación

4.1. Diseño y Generación de Datos

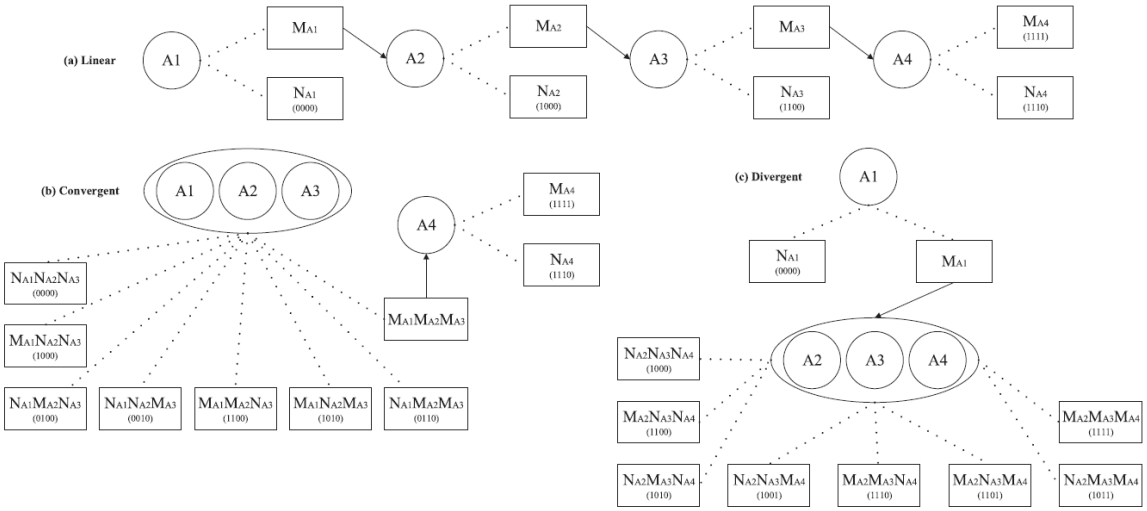
Se realizó un estudio de simulación considerando los factores de análisis y metodología sugerida por Zhan et al. (2020) y Zhan y He (2021) para evaluar si los parámetros propuestos en los modelos Hi-Long-DINA y Hi-Long-LLM pueden ser recuperados correctamente mediante distintas condiciones. Para implementarlo, se establecieron cuatro etapas. Primero, se determinó el tamaño de la muestra, el cual se fijó en 100 y 300 individuos (N = 100 y 300). Segundo, se definieron tres estructuras jerárquicas que se usarán: lineal, convergente y divergente. Tercero, se definió la cantidad de periodos donde se simularán las evaluaciones en 3 (T=3). Cuarto, se estableció el uso dos modelos de análisis: Hi-Long-DINA y Hi-Long-LLM. Las primeras tres etapas (cantidad de individuos, estructuras jerárquicas y cantidad de momentos) fueron aplicados para cada modelo de análisis.

Figura 4.1: Modelo secuencial lineal de atributos jerárquicos



Tomado de: Elaboración propia.

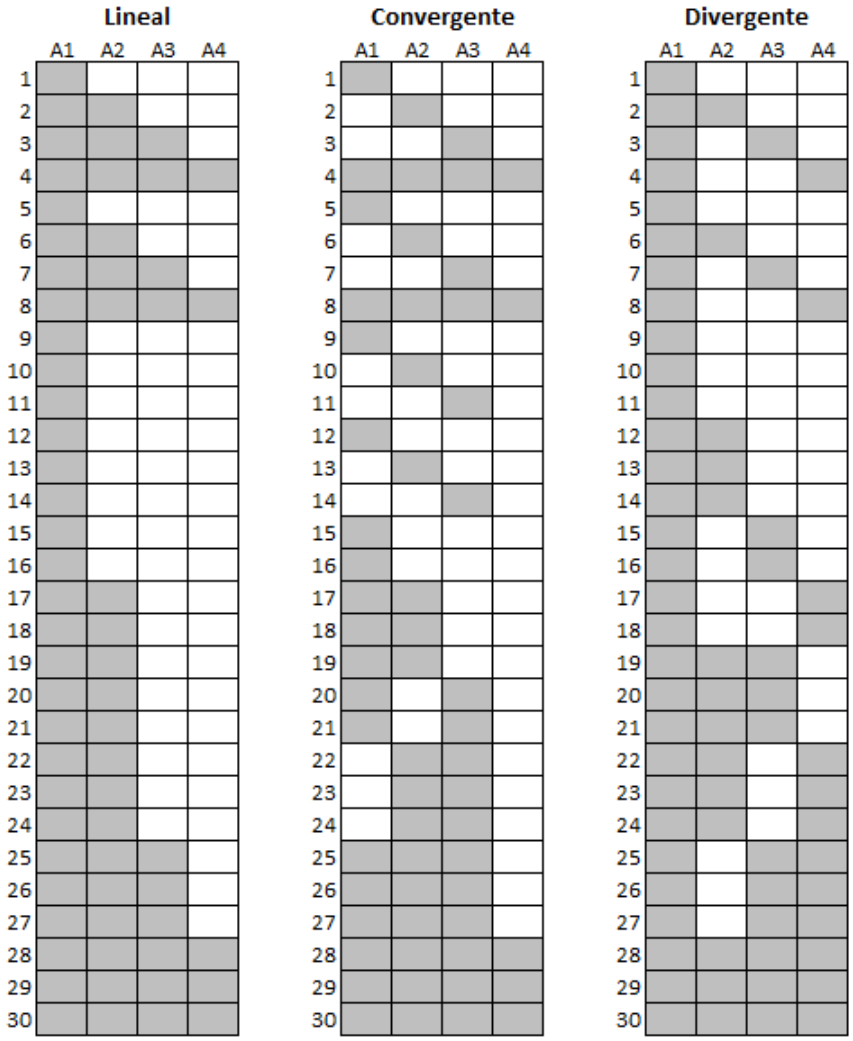
Figura 4.2: Árboles de dominio secuencial para simulación (π_{nct})



Tomado de: Zhan y He (2021).

Tomando como referencia la aplicación que se verá en el capítulo 5, se utilizó un diseño de evaluaciones paralelas para este estudio. Cada evaluación tuvo la misma matriz Q, conteniendo cuatro atributos (K=4) que fueron medidos por 30 ítems (I=30). Asimismo, considerando la estructura secuencial, no estuvo permitido considerar ítems que solo requerían los atributos de orden superior pero no incluían a los de orden inferior de acuerdo con la jerarquía correspondiente.

Figura 4.3: Matrices Q para simulación



Los recuadros grises indican cuáles atributos de la columna Ak, $k \in \{1, 2, 3, 4\}$, tienen que ser dominados por el individuo para responder correctamente al ítem de la fila i, $i \in \{1, 2, ..., 30\}$. Los recuadros blancos, cuáles no son requeridos.

Para el modelo Hi-Long-DINA, los parámetros de los ítems fueron generados de una distribución normal bivariada con una correlación negativa de coeficientes como se muestra:

$$\begin{pmatrix} \lambda_{0it} \\ \lambda_{1it} \end{pmatrix} \sim N\left(\begin{pmatrix} -2.197 \\ 4.394 \end{pmatrix}, \begin{pmatrix} 1 & -0.6 \\ -0.6 & 1 \end{pmatrix}\right), \tag{4.1}$$

donde $logit(x) = log\left(\frac{x}{1-x}\right)$.

Para el Hi-Long-LLM, los parámetros de los ítems fueron generados como se muestra:

$$\lambda_{0it} \sim N(-2.197, 1) \text{ y } \forall k, \lambda_{1ikt} \sim N\left(\frac{4.394}{\sum_{k=1}^{4} q_{ikt}}, 1\right). \tag{4.2}$$

Hubo una moderada correlación negativa entre los parámetros generados de adivinanza (como $\frac{exp(\lambda_{0it})}{1+exp(\lambda_{0it})}$) y desliz (como para Hi-Long-Dina $\frac{exp(\lambda_{0it}+\lambda_{1it})}{1+exp(\lambda_{0it}+\lambda_{1it})}$ o para Hi-Long-LLM $\frac{exp(\lambda_{0it}+\sum_{k=1}^{4}\lambda_{1ikt}\cdot q_{ikt})}{1+exp(\lambda_{0it}+\sum_{k=1}^{4}\lambda_{1ikt}\cdot q_{ikt})}$). Asimismo, se fijaron todos los parámetros estructurales latentes de segundo nivel: $\xi_k = 1.5$ (ξ fue constante para cada atributo); mientras que $\beta_k = (-1, -0.5, 0.5, 1)'$ teniendo un valor diferente para cada atributo (de modo que, la dificultad aumentaba para el siguiente).

Para las habilidades generales θ_n (tercer nivel), la correlaciones entre ellas se fijó en 0.9. Entre dos puntos de tiempo consecutivos, el crecimiento medio general ($\mu_t - \mu_{t-1}$) se fijó en 1 y la escala general de cambio ($\frac{\sigma}{\sigma_{t-1}}$) fue fijada en $\sqrt{1.25}$. Las habilidades generales en los T puntos de tiempo fueron generados de una distribución normal multivariada de acuerdo a la ecuación (3.1). En cada punto de tiempo, el perfil de atributo de cada individuo se generó usando la ecuación (3.2). Las respuestas a los ítems fueron generadas por una distribución Bernoulli(p_{nit}), donde p_{nit} fue mostrada en la ecuación (3.4) para el modelo Hi-Long-DINA y en la ecuación (3.5) para el Hi-Long-LLM. Se generaron un total de 30 conjunto de datos simulados[5].

4.2. Análisis

Se ejecutaron 15 replicaciones de los datos ajustándolos a los modelos Hi-Long-DINA y Hi-Long-LLM. En cada replicación, se aplicaron tres cadenas de Markov con puntos iniciales aleatorios. Cada cadena tuvo 15,000 iteraciones, de las cuales las primeras 7,500 fueron removidas (burn-in). La convergencia fue obtenida mediante el uso del PSRF ("potential scale reduction factor") menor a 1.2. Este indicador fue propuesto por Brooks y Gelman (1998), para monitorear la convergencia de las simulaciones iterativas comparando las varianzas entre múltiples cadenas. Los valores cercanos a 1 se consideran que han convergido a la distribución objetivo; mientras que los valores mayores a 1.2 implican que se debe continuar con la simulación porque no se ha llegado a la convergencia. Debido a la abundante cantidad de parámetros a estimar[6], 1680 cuando $N = 100$ y 4680 cuando $N = 300$, la convergencia puede tomar bastante tiempo y poder computacional; por eso, se decidió ser un poco más flexible en este indicador y trabajarlo como máximo a 1.2. En el Anexo E, se incluyen las gráficas de unas cadenas simuladas como medio visual para determinar la convergencia de

[5]Se simuló un conjunto de datos para cada punto de tiempo ($T = 3$) por cada una de las tres estructuras de atributos secuenciales (lineal, convergente y divergente): $3 \cdot 3 = 9$

[6]Los parámetros a estimar son: $g_{it} = I \cdot T$; $s_{it} = I \cdot T$; $\alpha_{nkt} = N \cdot K \cdot T$; $\theta_{nt} = N \cdot T$; donde N $\in \{100; 300\}$, I = 30, T = 3 y K = 4

las replicaciones.

La evaluación de la recuperación de los parámetros g, s y θ_t se calculó mediante el sesgo, la raíz del error cuadrático medio (RMSE) y la correlación (Cor) para cada modelo (Hi-Long-DINA y Hi-Long-LLM) y cada estructura jerárquica secuencial (convergente, divergente y lineal) entre los valores verdaderos (provenientes del conjunto de datos inicial) y los estimados de cada parámetro. Los valores de g y s tienen valores en un rango entre 0 y 1; mientras que los de θ pueden hallarse mayormente entre -3 y 3.

El sesgo de $\hat{\omega}$ fue calculado como:

$$Sesgo(\hat{\omega}) = \frac{\sum_{r=1}^{R}(\hat{\omega}_r - \omega_r)}{R} \tag{4.3}$$

La raíz del error cuadrático medio de $\hat{\omega}$ fue calculada como:

$$RMSE(\hat{\omega}) = \sqrt{\frac{\sum_{r=1}^{R}(\hat{\omega}_r - \omega_r)^2}{R}} \tag{4.4}$$

donde R (r = 1,...,R) es el número total de replicaciones realizadas, $\hat{\omega_r}$ es el valor estimado de los parámetro del modelo y ω_r es el valor verdadero de dichos parámetros. La correlación fue calculada comparando los valores verdaderos y los estimados de cada parámetro del modelo en una replicación.

Para evaluar la precisión de la recuperación de los atributos y perfiles de los indiviuos, se analizaron los ratios de clasificación de atributos (attribute correct classification rate, ACCR) y los ratios de clasificación de patrones (pattern correct classification rate, PCCR). El ACCR fue calculada como:

$$ACCR = \frac{\sum_{r=1}^{R}\sum_{n=1}^{N} I\{\hat{\boldsymbol{\alpha}}_{nkr} = \boldsymbol{\alpha}_{nkr}\}}{NR} \tag{4.5}$$

El PCCR fue calculada como:

$$PCCR = \frac{\sum_{r=1}^{R}\sum_{n=1}^{N} I\{\hat{\boldsymbol{\alpha}}_{nr} = \boldsymbol{\alpha}_{nr}\}}{NR} \tag{4.6}$$

donde R (r = 1,...,R) es el número total de replicaciones realizadas; N (n = 1,...,N) es el número de individuos; $\hat{\boldsymbol{\alpha}}_{nkr}$ y $\boldsymbol{\alpha}_{nkr}$ son el valor estimado y el valor real respectivamente del atributo k del individuo n de la replicación r e $I\{\cdot\}$ es una función indicadora. Los resultados del ACCR o el PCCR pueden variar desde 0 hasta 1. Los valores cercanos a 1 indican que el modelo recuperó adecuadamente todos los atributos dominados o perfiles de atributos del

individuo provenientes de los datos iniciales, respectivamente. Del modo inverso, los valores cercanos a 0 muestran que no se recuró ninguno de estos.

4.3. Resultados

Los modelos evaluados tuvieron mejores resultados recuperando los parámetros de los ítems (adivinanza g y desliz s), en los que se obtuvieron pequeñas desviaciones con respecto a los valores reales. No obstante, el modelo Hi-Long-LLM tuvo un bajo desempeño para recuperar las habilidades generales (ver Tabla 4.4) y clasificar los perfiles (ver Tabla 4.6). Asimismo, se noto que los resultados empeoraban para cada periodo siguiente (es decir, los resultados del periodo 3 fueron peores que los del periodo 2).

Tabla 4.1: Resultados de Parámetros de ítems estimados g y s del modelo Hi-Long-DINA

			Sesgo		RMSE		Cor	
T	N	Est Secuencial	g	s	g	s	g	s
3	100	Convergente	0.003	0.002	0.055	0.043	0.891	0.915
3	100	Divergente	0.004	0.003	0.050	0.043	0.898	0.912
3	100	Lineal	0.007	0.001	0.058	0.039	0.877	0.924
3	300	Convergente	0.002	-0.003	0.031	0.027	0.953	0.96
3	300	Divergente	0.002	0.002	0.034	0.027	0.949	0.965
3	300	Lineal	0.003	-0.001	0.035	0.026	0.951	0.966

Tabla 4.2: Resultados de Parámetros de ítems estimados g y s del modelo Hi-Long-LLM

			Sesgo		RMSE		Cor	
T	N	Est Secuencial	g	s	g	s	g	s
3	100	Convergente	0.022	-0.014	0.073	0.078	0.829	0.936
3	100	Divergente	0.060	-0.013	0.073	0.078	0.830	0.968
3	100	Lineal	0.075	-0.035	0.125	0.113	0.743	0.853
3	300	Convergente	0.012	-0.01	0.043	0.057	0.944	0.959
3	300	Divergente	0.067	0.001	0.106	0.028	0.809	0.992
3	300	Lineal	0.079	-0.029	0.127	0.090	0.780	0.907

En las Tablas 4.1 y 4.2, se muestran los resultados de las estimaciones a los parámetros de adivinanza g y desliz s hechas para los modelos Hi-Long-DINA y Hi-Long-LLM, respectivamente. Ambos modelos estimaron bien estos parámetros y la desviación generada fue muy pequeña. Se distingue que la estructura secuencial con mejores resultados es la convergente.

Tabla 4.3: Recuperación de Habilidades Generales θ_t del modelo Hi-Long-DINA

			Sesgo			RMSE			Cor		
T	N	Est Secuencial	θ_1	θ_2	θ_3	θ_1	θ_2	θ_3	θ_1	θ_2	θ_3
3	100	Convergente	-0.049	-0.503	-0.840	1.427	1.280	1.416	0.874	0.852	0.766
3	100	Divergente	0.054	-0.410	-0.783	1.546	1.285	1.356	0.854	0.828	0.763
3	100	Lineal	0.031	-0.410	-0.738	1.582	1.371	1.386	0.836	0.816	0.750
3	300	Convergente	-0.064	-0.553	-1.086	1.439	1.379	1.585	0.878	0.855	0.771
3	300	Divergente	0.005	-0.491	-0.935	1.488	1.346	1.458	0.851	0.827	0.775
3	300	Lineal	0.053	-0.490	-1.007	1.554	1.404	1.567	0.835	0.815	0.754

Tabla 4.4: Recuperación de Habilidades Generales θ_t del modelo Hi-Long-LLM

			Sesgo			RMSE			Cor		
T	N	Est Secuencial	θ_1	θ_2	θ_3	θ_1	θ_2	θ_3	θ_1	θ_2	θ_3
3	100	Convergente	-0.053	-0.552	-0.890	1.454	1.353	1.473	0.861	0.840	0.720
3	100	Divergente	0.048	-0.440	-0.760	1.597	1.378	1.425	0.828	0.801	0.701
3	100	Lineal	0.054	-0.410	-0.783	1.546	1.285	1.356	0.854	0.828	0.763
3	300	Convergente	-0.091	-0.607	-1.176	1.468	1.445	1.694	0.866	0.837	0.734
3	300	Divergente	-0.013	-0.621	-1.213	1.516	1.501	1.736	0.838	0.807	0.733
3	300	Lineal	0.044	-0.594	-1.312	1.607	1.578	1.911	0.802	0.762	0.654

En las Tablas 4.3 y 4.4, se muestran los θ_t recuperados para cada uno de los tres periodos de tiempo t. El sesgo y RMSE de ambos modelos tienen valores similares para cada periodo; por lo que, se observa que θ no es muy influenciado por las estructuras secuenciales ni por el modelo usado. El Hi-Long-DINA tiene unos resultados ligeramente mejores.

Tabla 4.5: Recuperación de Clasificación de Atributos estimados del modelo Hi-Long-DINA

			ACCR				
t	N	Est Secuencial	α_{1t}	α_{2t}	α_{3t}	α_{4t}	PCCR
1	100	convergente	0.996	0.989	0.994	0.875	0.856
2	100	convergente	0.998	0.994	0.994	0.861	0.850
3	100	convergente	0.998	0.997	0.996	0.888	0.880
1	100	divergente	0.995	0.906	0.947	0.954	0.857
2	100	divergente	0.996	0.918	0.952	0.969	0.885
3	100	divergente	0.999	0.959	0.972	0.974	0.944
1	100	lineal	0.999	0.930	0.917	0.891	0.795
2	100	lineal	0.999	0.934	0.903	0.874	0.784
3	100	lineal	0.999	0.958	0.928	0.872	0.826
1	300	convergente	1.000	0.990	0.997	0.884	0.874
2	300	convergente	1.000	0.997	0.997	0.863	0.858
3	300	convergente	0.998	0.999	0.994	0.878	0.873
1	300	divergente	0.997	0.911	0.945	0.961	0.865
2	300	divergente	0.994	0.920	0.946	0.964	0.883
3	300	divergente	0.998	0.951	0.963	0.973	0.937
1	300	lineal	0.999	0.925	0.911	0.881	0.783
2	300	lineal	1.000	0.924	0.919	0.871	0.782
3	300	lineal	1.000	0.952	0.928	0.883	0.833

Tabla 4.6: Recuperación de Clasificación de Atributos estimados del modelo Hi-Long-LLM

			ACCR				
t	N	Est Secuencial	α_{1t}	α_{2t}	α_{3t}	α_{4t}	PCCR
1	100	convergente	0.995	0.986	0.990	0.596	0.574
2	100	convergente	0.996	0.991	0.984	0.447	0.433
3	100	convergente	0.999	0.994	0.991	0.266	0.258
1	100	divergente	0.987	0.854	0.885	0.893	0.693
2	100	divergente	0.991	0.840	0.886	0.857	0.652
3	100	divergente	0.991	0.873	0.917	0.870	0.711
1	100	lineal	0.995	0.905	0.634	0.605	0.445
2	100	lineal	1.000	0.895	0.488	0.455	0.265
3	100	lineal	0.997	0.930	0.392	0.252	0.130
1	300	convergente	0.999	0.994	0.993	0.594	0.584
2	300	convergente	0.999	0.989	0.994	0.453	0.447
3	300	convergente	0.996	0.992	0.989	0.328	0.318
1	300	divergente	0.989	0.891	0.901	0.905	0.745
2	300	divergente	0.988	0.883	0.875	0.912	0.728
3	300	divergente	0.997	0.915	0.907	0.903	0.784
1	300	lineal	0.999	0.901	0.781	0.607	0.468
2	300	lineal	0.998	0.911	0.701	0.430	0.281
3	300	lineal	0.999	0.931	0.609	0.257	0.129

En las Tablas 4.5 y 4.6. se consolidan los resultados de los ACCR y PCCR para cada modelo con cada una de las tres estructuras secuenciales. Se muestran los ratios de clasificación en cada tiempo para cada atributo y perfil. Se puede apreciar que el modelo Hi-Long-DINA ofrece mejores resultados a nivel general. Dentro de este, la estructura divergente es la que mejores aciertos consiguió. En general, se ve un empeoramiento de los resultados para ambos modelos en el atributo 4 α_{4t} y en el cálculo del PCCR.

Capítulo 5

Aplicación a una prueba sobre operaciones matemáticas

En este capítulo, se presenta una aplicación de los modelos anteriormente descritos: Hi-Long-DINA y Hi-Long-LLM. Se utilizarán los datos de un experimento longitudinal elaborado por Tang y Zhan (2021) para analizar el efecto de diferentes tipos de retroalimentación en estudiantes de secundaria que están aprendiendo a realizar operaciones matemáticas de números racionales. Asimismo, se calculará la eficiencia de ambos modelos y se les comparará para ver cuál tiene el mejor ajuste con los datos reales. Los datos requeridos en este estudio empírico fueron facilitados amablemente por el doctor Peida Zhan.

5.1. Descripción de datos e instrumento utilizados

El estudio consistió en tomar evaluaciones paralelas en tres momentos a 276 alumnos de trece años en promedio. El tiempo transcurrido entre dos pruebas fue de una semana. Los participantes del experimento fueron divididos en tres grupos: de diagnóstico, tradicional y de control. Cada grupo tuvo 90, 92 y 94 estudiantes, respectivamente. El primero recibió tras cada evaluación un reporte de diagnóstico de retroalimentación (ver Figura 5.1, parte I y II) que contenía dos secciones: sus respuestas correctas e incorrectas y un estado del dominio de los atributos latentes. El segundo solo recibió un reporte tradicional con la corrección del examen donde se muestran las respuestas correctas e incorrectas (ver Figura 5.1, parte I). El tercero, de control, no recibió ningún reporte. Para este estudio, solo se trabajará con las respuestas del primer grupo de 90 estudiantes $N = 90$.

Figura 5.1: Reporte de Diagnóstico de Retroalimentación

Diagnostic Feedback Report

Student ID:

Part I. Your Answers

Multiple-Choice Items

Item	1	2	3	4	5	6	7	8	9	10	11	12
Answer Key	D	D	D	C	D	D	C	B	B	A	B	B
Your Answer	D	D	D	B	C	B	C	A	B	A	B	A
Accuracy	√	√	√	×	×	×	√	×	×	√	√	×

Calculation Items

Item	13	14	15	16	17	18
Correct Answer	21	−1	160	8/75	16	−13
Your Answer	21	−1	160	16/75	9	
Accuracy	√	√	√	×	×	×

Score: 9

Part II. Your Mastery Status of Knowledge Points

Knowledge Point	Mastery or Non-mastery	Probability*
A1: Rational Numbers	Yes	97%
A2: Related Concepts of Rational Numbers	Yes	94%
A3: Axis	No	11%
A4: Addition and Subtraction of Rational Numbers	Yes	76%
A5: Multiplication and Division of Rational Numbers	Yes	57%
A6: Mixed Operation of Rational Numbers	No	33%

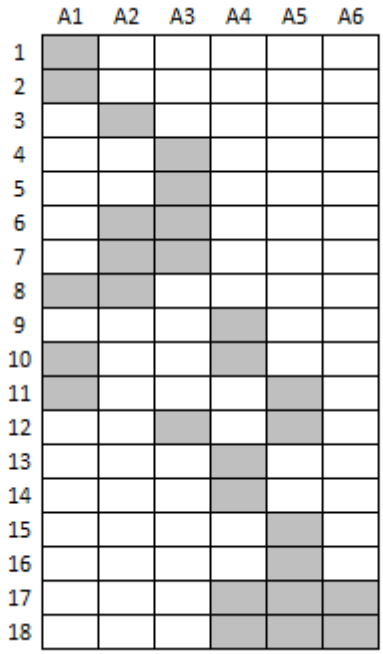
Note. * means the degree of certainty of classification.

Attribute Mastery Status: (A1, A2, A3, A4, A5, A6) = (110110)

Tomado de: Tang y Zhan (2021)

Los exámenes fueron diseñados usando la misma matriz Q (la cual se puede apreciar en la Figura 5.2), lo que garantiza la misma cantidad de ítems (18 preguntas con dificultades similares) y atributos latentes evaluados. Así, a pesar de tenerse ítems diferentes en cada prueba, estas son comparables longitudinalmente. Cada evaluación tuvo 18 ítems dicotómicos (que no fueron repetidos en otra prueba del experimento), 12 de opciones múltiples y 6 para realizar cálculos. Asimismo, se garantizó la confiabilidad y validación del instrumento mediante pruebas de monitoreo y control de calidad (Tang y Zhan (2021)).

Figura 5.2: Matriz Q para el estudio de aplicación



Los recuadros grises indican cuáles atributos de la columna Ak, $k \in \{1, 2, 3, 4\}$, tienen que ser dominados por el individuo para responder correctamente al ítem de la fila i, $i \in \{1, 2, ..., 30\}$. Los recuadros blancos, cuáles no son requeridos.

Solo se considerarán las respuestas del grupo de diagnóstico (primer grupo) de 90 estudiantes y sus respuestas a los tres exámenes en los $T = 3$ periodos de estudio, $t = (1, 2, 3)$. Más detalle sobre el proceso de desarrollo del instrumento utilizado puede obtenerse en Tang y Zhan (2020).

Como se aprecia en la Figura 5.2 y la Tabla 5.1 en este estudio se consideraran seis atributos ($K = 6$): A1, A2, A3, A4, A5 y A6; los cuales generan 40 perfiles permisibles (estados de conocimientos), $C^* = 40$. Los 18 ítems ($I = 18$) de las pruebas evaluaron el dominio de los estudiantes sobre estos atributos latentes los cuales fueron organizados en la estructura jerárquica observada en la Figura 5.3. Por un lado, los primeros tres atributos, A1, A2, A3, son independientes en la estructura mostrada; es decir, ninguno requiere de un dominio previo de otro atributo. Por otro lado, los atributos A4 y A5 son requisitos[7] para el atributo A6. El diagrama del proceso de dominio secuencial correspondiente (ver Figura 5.4) da un apoyo visual para identificar los perfiles de atributos permisibles.

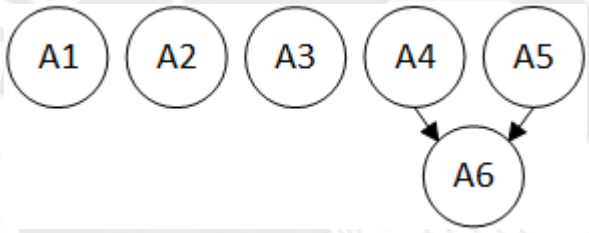
[7]Las flechas en la Figura 4.3 indican la dependencia unidireccional.

Tabla 5.1: Descripción de Atributos evaluados

Abrev.	Atributo	Contenido del atributo
A1	Números Racionales	Conceptos y clasificaciones
A2	Conceptos relacionados a los números racionales	Números opuestos. Valor absoluto
A3	Recta Real de números	Conceptos, conversión de números, comparación del tamaño de los números.
A4	Adición y sustracción de números racionales	Adición, sustracción y sus reglas operativas
A5	Multiplicación y división de números racionales	Multiplicación, división y sus reglas operativas. Involución, negación y reciprocidad. Reducción de fracciones a denominadores comunes.
A6	Operaciones combinadas de números racionales	Primero, operar involuciones, luego, multiplicación y división; finalmente, adición y sustracción. Si hay números entre paréntesis, se comienzan con estos.

Adaptado de: Tang y Zhan (2021).

Figura 5.3: Jerarquía de Atributos de la evaluación



Tomado de: Elaboración propia.

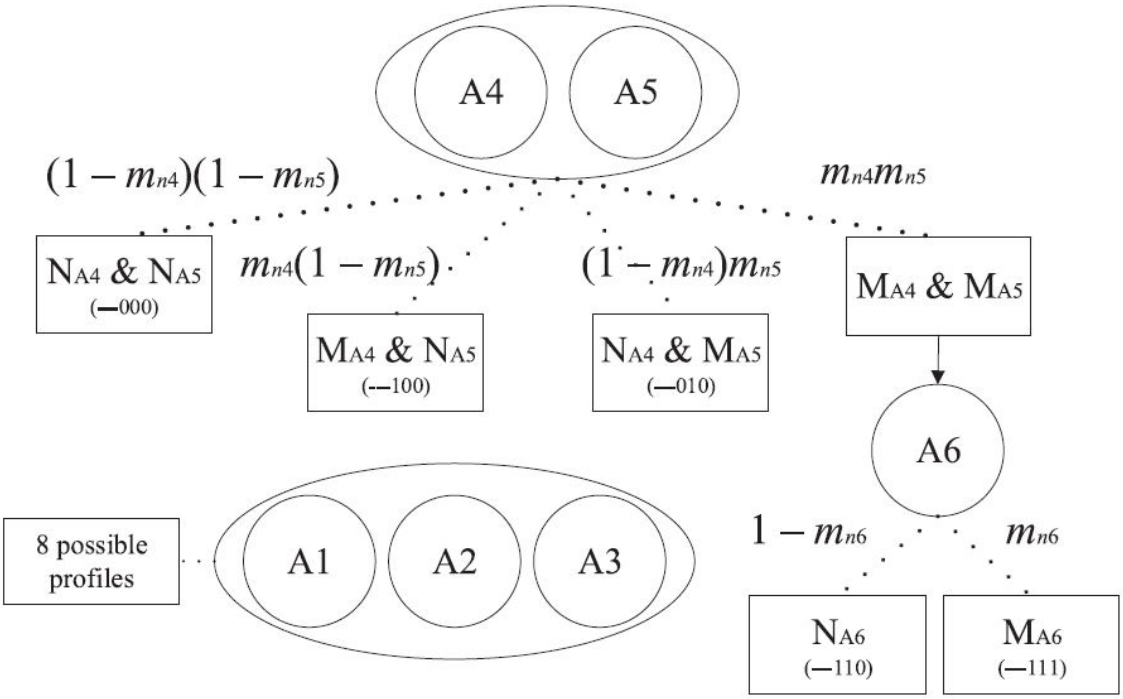
La estructura jerárquica de los atributos latentes propuesta en la Figura 5.3 genera 40 perfiles de atributos permisibles[8], $C^* = 40$, en vez de 64 (2^6), que sería la cantidad utilizada por los modelos anteriores que no consideran la dependencia entre atributos, sino que aceptan todas las combinaciones. Por ejemplo, tras observar que los atributos A4 y A5 son necesarios para la existencia de A6, es imposible proponer al patrón "000101" (que indica que el individuo domina A4 y A6, pero no, A5). Dicho patrón sí se tomaría como posible si se trabaja con las 64 posibilidades, lo que generaría inconsistencia en el cálculo total de las probabilidades.

La forma cómo se generan estos perfiles puede ser apreciada en la Figura 5.4. Primero, tenemos las 8 combinaciones posibles entre A1, A2 y A3. Luego tenemos 3 combinaciones

[8]Si el individuo domina alguna combinación de A1, A2 y A3, existen 8 posibilidades (2^3). Luego existen 4 escenarios donde se considerán esas 8 posibilidades: dominar solo A4; solo A5; A4 y A5 pero no A6; dominar A4, A5 y A6. De esta forma, tendríamos $2^3 + 4.(2^3) = 40$ perfiles permisibles.

posibles entre A4 y A5 de no dominar ambas a la vez (...000, ...100, ...010) y dos posibilidades de sí hacerlo (...110 y ...111). Estas últimas dos incluye una de dominar A6 y otra de no hacerlo. Finalmente, si incluimos los 8 perfiles posibles para cada combinación de A4, A5 y A6, obtenemos las 40 combinaciones posibles ilustradas en el diagrama de la Figura 5.4.

Figura 5.4: Proceso de dominio secuencial de atributos de estudio empírico



Adaptado de: Zhan y He (2021).

5.2. Análisis

Para analizar los modelos presentados, Hi-Long-DINA y Hi-Long-LLM, se realizaron comparaciones contra dos modelos presentados en el capítulo 2: sLong-DINA y sLong-LLM (comentados en el capítulo 2). La diferencia principal entre ambos pares de modelos radica en que unos consideran las estructuras secuenciales de sus atributos y los otros no. Los primeros solo consideran perfiles permisibles ($C^* = 40$) generados por las estructuras secuenciales. En contraste, los segundos no modelan ninguna dependencia que pudiera ocurrir (como las presentadas en la Figura 2.2), permitiendo la existencia de perfiles imposibles. De esta manera, consideran 64 ($C = 64 = 2^6$) perfiles totales, en vez de los 40 permisibles.

Los cuatro modelos, Hi-Long-DINA, sLong-DINA, Hi-Long-LLM y sLong-LLM, fueron

aplicados para ajustar los datos recogidos en las evaluaciones. Se usaron tres cadenas de Markov con 15,000 iteraciones cada una, de las cuales las primeras 7,500 fueron removidas (burn-in) y se uso un intervalo de adelgazamiento de 5 (*thinning*). La convergencia fue diagnosticada mediante el uso del PSRF ("potential scale reduction factor") menor a 1.2 (Brooks y Gelman (1998)) debido a la gran cantidad de parámetros que deben ser también estimados en este estudio[9].

Asimismo, se empleo el criterio del "Posterior predictive model checking" (PPMC) elaborado por Gelman et al. (2013) para medir el ajuste general del modelo. Un valor de probabilidad predictiva posterior (ppp) cercano a 0.5 indica que no hay diferencias sistemáticas entre los valores reales y estimados; por tanto, es un ajuste adecuado para el modelo. El uso del Deviance Information Criterion (DIC, propuesto por Spiegelhalter et al. (2002)) fue usado para la selección de modelos[10].

5.3. Resultados

Tabla 5.2: Comparación de DIC de cada modelo

Modelo	DIC	ppp
Hi-Long-DINA	4293.08	0.415
sLong-DINA	4305.41	0.384
Hi-Long-LLM	4323.04	0.628
sLong-LLM	4360.71	0.654

En la tabla 5.2. se muestra el ajuste de cada modelo. Se puede observar que los Hi-Long-DINA y sLong-DINA han conseguido un mejor ajuste que los LLM. Asimismo, se identifica que los modelos presentados, que incluyen las estructuras secuenciales en su formulación, tuvieron un mejor resultado que sus contrapartes que nos lo hacen. Por otro lado, el indicador ppp muestra que los modelos no tienen problemas para ajustarse a los datos, sin llegar a sobreajustarse.

[9]Los parámetros a estimar por modelo son 11,178: $g_{it} = I \cdot T$; $s_{it} = I \cdot T$; $\pi_{nC^*t} = N \cdot C^* \cdot T$; $\theta_{nt} = N \cdot T$; donde N = 90, I= 18, T = 3, C^*= 40 y K = 6.

[10]Un menor valor del DIC indica un mejor ajuste del modelo a los datos.

Tabla 5.3: Resultados de Proporciones combinadas con jerarquías secuenciales

Perfiles	Hi-Long-DINA			sLong-DINA			Hi-Long-LLM			sLong-LLM		
Permisibles	*t=1*	*t=2*	*t=3*	*t=1*	*t=2*	*t=3*	*t=1*	*t=2*	*t=3*	*t=1*	*t=2*	*t=3*
000000	0.10	0.11	0.05	0.09	0.08	0.04	0.08	0.09	0.08	0.10	0.07	0.06
100000	0.10	0.08	0.07	0.07	0.06	0.04	0.13	0.11	0.10	0.14	0.11	0.11
010000	0.02	0.02	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
110000	0.08	0.06	0.06	0.04	0.04	0.03	0.09	0.07	0.06	0.08	0.07	0.08
101000	0.01	0	0	0	0	0	0	0	0	0	0	0
111000	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
100100	0.02	0.02	0.02	0.01	0.01	0.01	0.02	0.02	0.01	0.02	0.02	0.02
110100	0.05	0.04	0.06	0.02	0.02	0.03	0.07	0.06	0.06	0.05	0.05	0.05
101100	0.01	0.01	0.01	0	0	0	0	0	0	0	0	0
111100	0.05	0.03	0.05	0.01	0.01	0.02	0.07	0.05	0.05	0.04	0.03	0.03
111010	0.01	0	0.01	0	0	0	0	0	0	0	0	0
110110	0.01	0	0.01	0.01	0	0.01	0.02	0.01	0.01	0.02	0.02	0.01
111110	0.03	0.02	0.03	0.03	0.02	0.03	0.12	0.11	0.10	0.13	0.11	0.11
110111	0.02	0.01	0.02	0.02	0.01	0.02	0.02	0.01	0.01	0.01	0.01	0.01
101111	0.01	0.01	0.01	0.01	0	0.01	0	0	0	0	0	0
111111	0.45	0.55	0.55	0.46	0.54	0.56	0.32	0.42	0.47	0.34	0.43	0.44
Otros	0.02	0.03	0.03	0	0	0	0.03	0.03	0.03	0	0	0
Suma	1.00	1.00	1.00	0.79	0.81	0.82	1.00	1.00	1.00	0.95	0.94	0.94

La Tabla 5.3, presenta la proporción combinada para cada modelo analizado en cada periodo t de los cuarenta perfiles de atributos permisibles o estados de conocimientos ($C^* = 40$). Se muestran las proporciones que sean mayores o iguales a 0.01 de probabilidad de ocurrencia; en caso contrario, se han agrupado en la fila "Otros". En el Anexo E, se muestran todas las proporciones combinadas, de los perfiles permisibles y los imposibles. Los perfiles imposibles se han quitado del análisis y, por tanto, no se consideran en la tabla.

Se puede apreciar que, al retirar los perfiles imposibles, la suma de los modelos "sLong" no suma 1. Lo que implica que ambos modelos han clasificado a uno o más individuos con perfiles imposibles. En particular, el modelo que ha categorizado peor a los individuos es el sLong-DINA. También, se observa el desplazamiento de los perfiles más básicos (que tienen dominio de pocos atributos, como "000000" o "100000") en los primeros periodos hacia los perfiles más complejos (donde se dominan la mayoría de los atributos, como "111111") en el tercer periodo. La mejora del estado de conocimiento de los individuos se aprecia en los cuatro modelos.

Por el lado de los parámetros g y s de los ítems, encontramos similitud en la tendencia entre las estimaciones realizadas de ambos modelos. Conforme pasan los periodos, el paráme-

tro de adivinanza suele aumentar y el de desliz tiende a disminuir. También, notamos que los picos del parámetros de desliz suelen aparecer cada vez en una pregunta más avanzada.

Tabla 5.4: Comparación de estimaciones para el parámetros de adivinanza g_{it}

	Hi-Long-DINA			sLong-DINA			Hi-Long-LLM			sLong-LLM		
Ítem	*t=1*	*t=2*	*t=3*	*t=1*	*t=2*	*t=3*	*t=1*	*t=2*	*t=3*	*t=1*	*t=2*	*t=3*
1	0.18	0.60	0.52	0.18	0.59	0.51	0.17	0.64	0.59	0.14	0.59	0.48
2	0.15	0.22	0.14	0.15	0.21	0.14	0.21	0.30	0.41	0.22	0.26	0.30
3	0.03	0.37	0.42	0.03	0.36	0.37	0.05	0.50	0.61	0.05	0.49	0.61
4	0.37	0.36	0.59	0.37	0.36	0.58	0.44	0.41	0.64	0.43	0.40	0.65
5	0.09	0.18	0.33	0.09	0.18	0.33	0.11	0.21	0.35	0.12	0.20	0.35
6	0.07	0.13	0.39	0.07	0.13	0.41	0.06	0.08	0.38	0.06	0.07	0.37
7	0.30	0.37	0.54	0.29	0.37	0.52	0.07	0.09	0.35	0.09	0.09	0.35
8	0.23	0.40	0.60	0.22	0.40	0.57	0.09	0.18	0.45	0.10	0.17	0.38
9	0.14	0.28	0.20	0.14	0.27	0.15	0.18	0.30	0.31	0.19	0.30	0.30
10	0.06	0.16	0.26	0.06	0.16	0.26	0.04	0.09	0.17	0.04	0.08	0.15
11	0.24	0.51	0.58	0.25	0.51	0.57	0.08	0.41	0.28	0.08	0.40	0.24
12	0.10	0.09	0.35	0.10	0.08	0.35	0.10	0.08	0.34	0.10	0.08	0.34
13	0.08	0.06	0.24	0.09	0.05	0.18	0.11	0.09	0.44	0.12	0.08	0.44
14	0.10	0.09	0.28	0.10	0.08	0.23	0.12	0.12	0.47	0.12	0.12	0.46
15	0.03	0.07	0.13	0.03	0.07	0.13	0.05	0.12	0.19	0.05	0.10	0.19
16	0.02	0.02	0.02	0.02	0.01	0.02	0.03	0.03	0.03	0.03	0.03	0.03
17	0.06	0.03	0.05	0.07	0.02	0.05	0.04	0.02	0.04	0.02	0.02	0.03
18	0.01	0.02	0.10	0.01	0.02	0.10	0.01	0.02	0.04	0.01	0.01	0.04

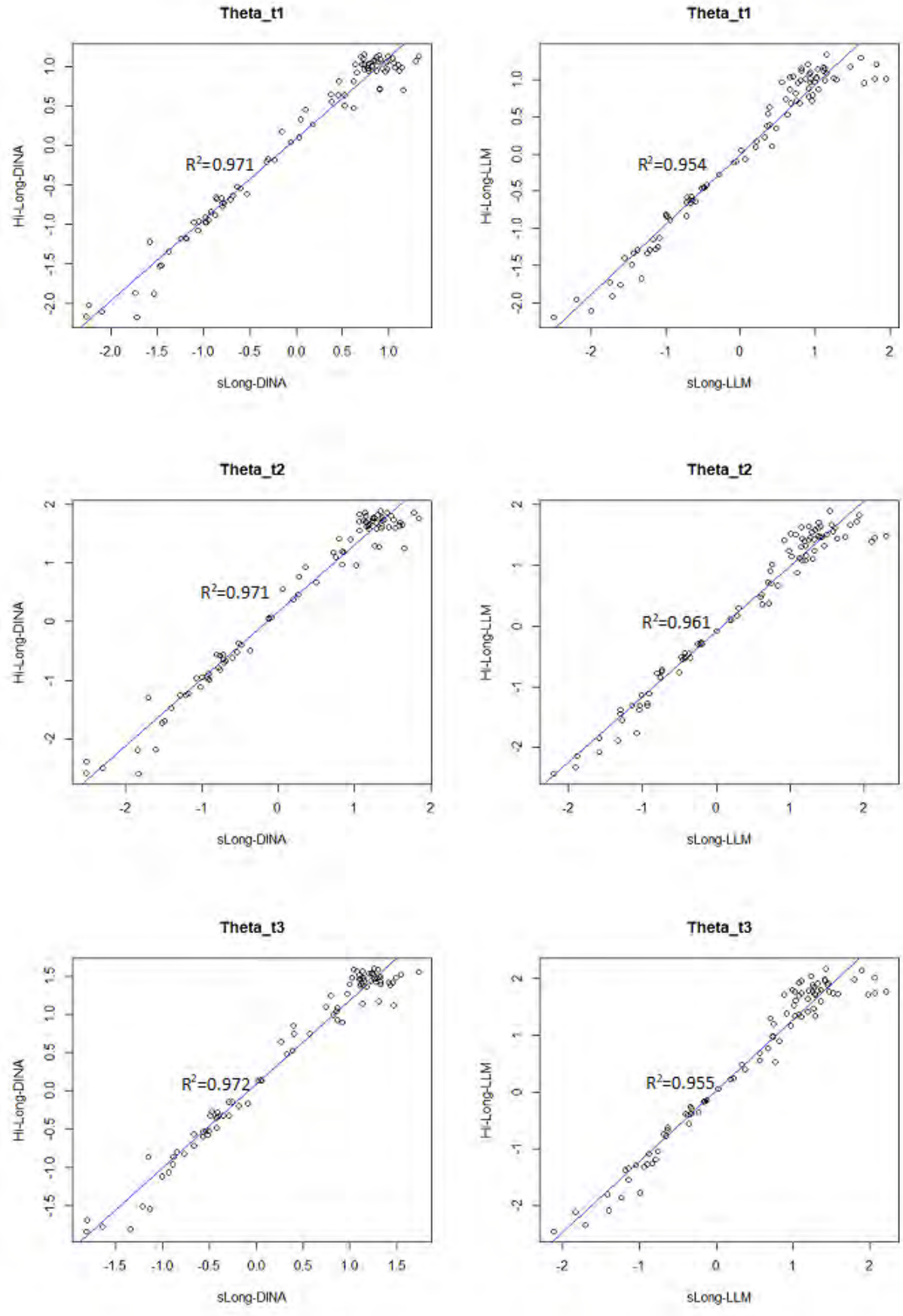
Tabla 5.5: Comparación de estimaciones para el parámetros de desliz s_{it}

	Hi-Long-DINA			sLong-DINA			Hi-Long-LLM			sLong-LLM		
Ítem	*t=1*	*t=2*	*t=3*	*t=1*	*t=2*	*t=3*	*t=1*	*t=2*	*t=3*	*t=1*	*t=2*	*t=3*
1	0.16	0.07	0.04	0.16	0.07	0.04	0.18	0.08	0.05	0.17	0.08	0.05
2	0.05	0.03	0.03	0.05	0.03	0.03	0.08	0.05	0.05	0.08	0.06	0.06
3	0.38	0.03	0.01	0.38	0.03	0.01	0.41	0.08	0.02	0.40	0.08	0.02
4	0.06	0.01	0.01	0.07	0.01	0.01	0.08	0.02	0.01	0.07	0.02	0.02
5	0.31	0.34	0.22	0.32	0.34	0.23	0.29	0.35	0.21	0.30	0.35	0.21
6	0.90	0.58	0.38	0.90	0.59	0.38	0.89	0.59	0.38	0.89	0.59	0.38
7	0.15	0.07	0.01	0.16	0.07	0.01	0.17	0.07	0.02	0.17	0.09	0.01
8	0.12	0.14	0.05	0.12	0.14	0.05	0.14	0.13	0.05	0.15	0.14	0.06
9	0.04	0.14	0.11	0.04	0.13	0.12	0.06	0.14	0.10	0.06	0.14	0.09
10	0.64	0.61	0.34	0.63	0.61	0.35	0.64	0.61	0.34	0.64	0.61	0.33
11	0.07	0.08	0.02	0.07	0.08	0.02	0.08	0.08	0.03	0.08	0.09	0.02
12	0.41	0.79	0.59	0.41	0.79	0.60	0.41	0.79	0.58	0.41	0.79	0.58
13	0.16	0.02	0.03	0.17	0.02	0.03	0.17	0.02	0.04	0.17	0.03	0.04
14	0.09	0.09	0.05	0.08	0.09	0.05	0.09	0.11	0.07	0.09	0.10	0.06
15	0.35	0.11	0.04	0.34	0.11	0.04	0.30	0.12	0.04	0.31	0.12	0.04
16	0.81	0.61	0.41	0.80	0.61	0.41	0.80	0.60	0.39	0.81	0.61	0.39
17	0.05	0.30	0.06	0.04	0.30	0.06	0.04	0.26	0.04	0.03	0.23	0.05
18	0.69	0.20	0.13	0.69	0.20	0.14	0.58	0.10	0.10	0.62	0.13	0.08

En las Tablas 5.4 y 5.5, se muestran respectivamente los valores recuperados de los parámetros de adivinanza y desliz. Podemos notar que los modelos de las mismas familias (Hi-Long-DINA y sLong-DINA; Hi-Long-LLM y sLong-LLM) tienen valores muy cercanos en cada ítem y periodo de tiempo. De manera preliminar, podríamos indicar que las estructuras secuenciales no influyen sobre las estimaciones de los parámetros de los ítems; debido a que, como se ven en las tablas, los valores recuperados son similares para ambos modelos, sea que use o no las estructuras secuenciales.

La Figura 5.5 muestra diagramas de dispersión donde se observa la correlación entre las habilidades generales estimadas para cada periodo por los modelos similares. Se aprecia que existe una alta correlación entre ambas estimaciones de los modelos DINA y LLM. Además, esta pareciera aumentar conforme el tiempo transcurre.

Figura 5.5: Diagrama de dispersión para correlación entre θ_t de modelos similares



Capítulo 6

Conclusiones

La educación es un proceso complejo pues requiere considerar múltiples factores para encontrar los mejores caminos de enseñanza y guiar a los estudiantes a través de estos de la manera más efectiva. Frente a esta complejidad, se hace necesario el empleo de modelos de diagnóstico cognitivo para diseñar procesos de aprendizaje adecuados que garanticen una educación de calidad. Este trabajo ha propuesto dos modelos de diagnóstico cognitivo jerárquico secuenciales en evaluaciones longitudinales para brindar herramientas de diagnóstico y desarrollo de trayectorias de aprendizaje que maximicen las probabilidades de dominio de habilidades o capacidades requeridas.

Los modelos presentados, Hi-Long-DINA y Hi-Long-LLM, consideran una jerarquía de habilidades organizada en tres niveles, de la general a la específica, donde se pueden establecer relaciones de dependencia entre los atributos latentes, que generan ciertos perfiles posibles, y evalúan en el tiempo el desarrollo del individuo. Esto permite identificar su estado de conocimiento y guiarlo a través del proceso de aprendizaje más adecuado dada su probabilidad de desempeño.

Un estudio de simulación fue realizado utilizando ambos modelos y tres estructuras jerárquicas secuenciales para comparar sus capacidades de recuperación de parámetros y categorización de perfiles. El Hi-Long-DINA tuvo mejores resultados que el Hi-Long-LLM para recuperar los parámetros iniciales y clasificar los estados de conocimientos. Mientras que el primero mantuvo sus resultados en la mayoría de los tiempos, el segundo tuvo una degradación conforme pasaban los periodos de tiempo. Este resultado coincide con lo mencionado por Zhan y He (2021) sobre la sensibilidad que tiene el modelo Hi-Long-LLM a la jerarquía secuencial de atributos; lo que no sucede con el Hi-Long-DINA que mantiene buenos

resultados independientemente de la estructura secuencial usada. Igualmente, la regla compensatoria del LLM puede verse afectada negativamente cuando se incorpora una secuencia ordenada de atributos requeridos. Finalmente, se vio que, en ambos modelos, la estructura secuencial divergente consiguió los mejores resultados.

Un estudio de aplicación también fue desarrollado para evaluar la capacidad de diagnóstico de los modelos propuestos y compararlos con otros similares que no consideran las estructuras jerárquicas secuenciales. Los resultados demuestran que ambos modelos presentados en la tesis lograron identificar los perfiles de los individuos en una evaluación longitudinal considerando una estructura secuencial previamente diseñada por expertos. Asimismo, estos diagnosticaron el estado del conocimiento de los individuos solo en los perfiles posibles de existir; a diferencia de los otros modelos contra los que fueron comparados, quienes clasificaron algunas veces a los evaluados en perfiles imposibles de existir. También, se pudo observar que el ajuste de los modelos (evaluado por el DIC) presentados fue mejor para aquellos que consideraron las estructuras jerárquicas secuenciales que para los que no lo hicieron.

A pesar que este tipo de modelos pueden ser útiles para diagnosticar los estados de conocimientos de los individuos, existen limitaciones que serían aconsejables abordar en próximas investigaciones. En primer lugar, el modelo no contempla la evaluación de la idoneidad de las estructuras jerárquicas secuenciales. Asume que esta es correcta y formula las dependencia de los atributos con base en esta jerarquía. Una jerarquía secuencial mal diseñada puede generar errores de cálculo y fallas al clasificar los estados de conocimiento de los estudiantes. Sería recomendable que en futuros trabajos se pueda incluir algún método para detectar cuando una estructura secuencial este mal planteada.

En segundo lugar, los modelos presentados en este trabajo no han tomado en cuenta aún la posibilidad de introducir covariables (en su primer nivel) que pudieran afectar a este. Estos efectos pueden ser fijos, dada la consideración de otros factores que pudieran influenciar en las respuestas de los individuos, o aleatorios, por ejemplo, si las pruebas tienen ítems de anclaje o estructuras testlets que violan la independencia local. Un avance en este último sentido fue considerado en Zhan et al. (2019).

En tercer lugar, se pudo observar que esta clase de modelos tiene un desafío grande para llegar a la convergencia de los parámetros debido a que suelen contar con una gran cantidad de estos. Para asegurar la credibilidad de las estimaciones, en esta tesis, se usaron como criterios de información del modelo: DIC, PSRF, ppp y gráficas de mcmc (donde se ven la

densidad, la media por iteración y la correlación). Sin embargo, sería muy útil identificar nuevos criterios que garanticen la convergencia sin requerir tanta capacidad computacional. Del mismo modo, se podría investigar cuáles otros algoritmos de muestreo, como STAN, HMC o Langevin, podrían acelerar la convergencia de las simulaciones. Esto sería muy importante si los parámetros a estimar llegasen a incrementar.

En cuarto lugar, solo fueron presentados como modelos de medición (primer nivel) los basados en el DINA y LLM. Sin embargo, la versatilidad de la estructura jerárquica permite modificarla usando otros diferentes, que podrían ser más adecuados para ciertas finalidades, como el rRUM o el LCDM. A pesar de ello, sigue siendo necesario un investigar detalladamente el desempeño de estas nuevas variantes del modelo.

En quinto lugar, casi todos los MDCL asumen la invarianza de su estructura latente (la cual implica que la estructura latente entre sus atributos no cambia a través del tiempo, de un periodo a otro). Debido a que la mayoría de evaluaciones longitudinales de diagnóstico congnitivo no se desarrollan durante un largo periodo de tiempo, se asume que la estructura latente es invariante. Por eso, sería importante analizar si la hipótesis de invarianza de las estructuras latentes longitudinales es correcta para las evaluaciones cognitivas de amplios periodos de tiempo.

Por todo lo comentado anteriormente, podemos observar que los modelos de diagnóstico cognitivo longitudinal que consideran las estructuras jerárquicas de atributos secuenciales, a pesar de sus limitaciones, se acercan más fidedignamente a la realidad de los estudiantes que otros modelos más antiguos. Su uso permite la clasificación del estado de conocimiento de un individuo sobre un tema, identificando cuáles capacidades o habilidades domina y cuáles no, pudiendo entender sus debilidades y fortalezas. A la vez, permite diseñar trayectorias de aprendizaje personalizadas al estadío cognitivo del estudiante, evitando que se le enseñen cosas que ya sabe y reforzando aquello que requerido para avanzar en su aprendizaje. Naturalmente, existen limitaciones de los modelos presentados; no obstante, se comentan algunas de estas para que puedan ser estudiadas en próximas investigaciones.

Capítulo 7

Anexos

7.1. Anexo A: Script en R de modelo para Estudio de Simulación

```
# Ejercicio de Simulacion - Tesis
# Nota1: Actualmente este script corre una simulación con el modelo Hi-Long-LLM
# y la estructura Convergente. Para cambiarlo se debe modificar las
# variables "mm" (donde: 1=Hi-Long-DINA, 2=Hi-Long-LLM)
# y "es_j" (donde: 1=lineal, 2=convergente, 3=divergente)

##### Inicio de parte I: Generacion de datos input para simulación  #####

# Limpiar y fijar carpeta
rm(list = ls())
setwd(dirname(rstudioapi::getActiveDocumentContext()$path))

# Nota2: Descomentar para instalar librerías (con: Ctrl + Shift + c)
# 1. Instalación y Carga de librerias
# suppressMessages(install.packages("dplyr"))
# suppressMessages(install.packages("MASS"))
# suppressMessages(install.packages("tidyverse"))
# suppressMessages(install.packages("statip"))
# suppressMessages(install.packages("MCMCglmm"))
# suppressMessages(install.packages("R2jags"))
# suppressMessages(install.packages("MCMCvis"))
```

```
# suppressMessages(install.packages("coda"))
# suppressMessages(install.packages("msm"))
# suppressMessages(install.packages("faux"))

suppressMessages(library(dplyr))
suppressMessages(library(MASS))
suppressMessages(library(tidyverse))
suppressMessages(library(statip))
suppressMessages(library(MCMCglmm))
suppressMessages(library(R2jags))
suppressMessages(library(MCMCvis))
suppressMessages(library(coda))
suppressMessages(library(msm))
suppressMessages(library(faux))



# 2.Preparación de datos

# Definición de factores de simulación
N = 100 # individuos en muestra (loop n);
es_j  <- 2 #estructura jerarquica que se usará: 1=lineal, 2=convergente, 3=divergente
T <- 3 # periodos de evaluación (loop t)
mm <- 2 #modelo de medicion: 1=Hi-Long-DINA, 2=Hi-Long-LLM

# Carga de inputs: Matriz Q, Perfiles Permisibles
##indicar folder donde se almacenarán los datos

load("simulacion_inputs.RData") #carga de matrices Q y perfiles permisibles

## Carga matriz Q
sim.matriz_Q_lista <- list(sim.matrizQ_lineal, sim.matrizQ_convergente,
                           sim.matrizQ_divergente)
matriz_Q <- sim.matriz_Q_lista[[es_j]]
```

```
## Carga atributos permisibles
sim.atributos_permisibles_lista <- list(sim.atributos_permisibles_lineal,
                                        sim.atributos_permisibles_convergente,
                                        sim.atributos_permisibles_divergente)
atributos_permisibles <- sim.atributos_permisibles_lista[[es_j]]

# Definición de Parámetros de ítems y atributos
SI = 30 #Número de ítems en cada punto de tiempo (loop i);
K = 4 #Número de atributos (loop k);
C <- dim(atributos_permisibles)[1] #Número de perfiles permisibles

# Revisión
mod_medicion <- case_when(mm == 1 ~ "HiLongDINA", mm == 2 ~ "HiLongLLM")
est_jerarq <- case_when(es_j == 1 ~ "lineal", es_j == 2 ~ "convergente",
                        es_j == 3 ~ "divergente")

print(paste("Parámetros de Simulación: ", "Tamaño de Muestra:",N,"|| N. Items:",
            SI,"|| N. Atributos:",K,"|| Tiempos:",T,"|| Estructura Jerarquica:",
            est_jerarq,"|| Modelo de Medición:", mod_medicion ,
            "|| N. Patrones Permitidos:",C))

# 3. Generación de datos

### Parametro de tercer nivel
theta_mu <- seq(0, by=1, length.out=3) # crecimiento promedio de mu en +1 cada t

#Generación de parámetros de Theta
sigma_th <- 1
theta_sigma <- matrix(c(sigma_th, 0.9, 0.9,
                        0.9, sqrt(1.25)*sigma_th, 0.9,
                        0.9, 0.9, sigma_th*1.25), nrow=3, byrow=T)
theta_omega <- chol2inv(chol(theta_sigma))

### Generación de Theta
```

```
f.theta <- function(theta_mu, theta_omega){
  mvrnorm_generados <- as.data.frame(matrix(NA, nrow=N, ncol=T))
  for(a in 1:N){
    mvrnorm_generados[a,] <- mvrnorm(n=1, mu=theta_mu, Sigma=theta_omega)
  }
  mvrnorm_generados
}
theta_nt <- f.theta(theta_mu, theta_omega)

### Parametros de segundo nivel
xi_k <- c(1.5, 1.5, 1.5, 1.5)
beta_k <- c(-1, -0.5, 0.5, 1)

### Calculo de m_nkt:
f.m_nkt <- function(theta, xi, beta){
  m_nkt <- array(data=NA, dim=c(N, K ,T))
  for (n in 1:N){
    for (t in 1:T){
      for (k in 1:K){
        m_nkt[n,k,t] <- (exp(xi[k]*theta_nt[n,t] - beta[k]))/(1+(exp(xi[k]
        *theta_nt[n,t] - beta[k])))
      }
    }
  }
  m_nkt
}
m_nkt <- f.m_nkt(theta_nt, xi_k, beta_k) #; head(m_nkt)

### Parametros de primer nivel
### Generación de alpha
alpha <- array(data=NA, dim=c(N, K, T))

for (t in 1:T){
  for (n in 1:N){
```

```
    for (k in 1:K){
      alpha[n,k,t] <- rbern(n=1, prob=m_nkt[n,k,t])
    }
  }
}

### Lambdas para Hi-Long-DINA
dina_meanvector <- c(-2.197, 4.394)
dina_covariance.matrix <- matrix(c(1, -0.6, -0.6, 1), ncol = 2)

### función para creación de Lambdas para DINA
f.lambdas_dina <- function(dina_mu, dina_sigma){
  lambda0 <- matrix(NA, nrow = SI, ncol = T);
  lambda1 <- matrix(NA, nrow = SI, ncol = T);
  for (t in 1:T) {
    new_lambda <- mvrnorm(n = SI, mu = dina_mu, Sigma = dina_sigma)
    for (i in 1:SI) {
      lambda0[i,t] <- new_lambda[i,1]
      lambda1[i,t] <- new_lambda[i,2]
    }
  }
  list(lambda0_dina=lambda0, lambda1_dina=lambda1)
}

## Generación de Lambdas para modelo Hi-Long-DINA
lambda_dina <- f.lambdas_dina(dina_meanvector, dina_covariance.matrix)
lambda0_dina <- lambda_dina$lambda0_dina
lambda1_dina <- lambda_dina$lambda1_dina

## Lambdas PARA Hi-Long_LLM
param_llm_lambda0 <- c(-2.197, 1)
param_llm_lambda1 <- c(4.394, 1)

### función de creación de Lambdas para LLM
```

```
f.lambdas_llm <- function(param_llm_lambda0, param_llm_lambda1){
  lambda0 <- matrix(NA, nrow = SI, ncol = T)
  lambda1 <- array(data=NA, dim=c(SI, K, T))
  for (t in 1:T) {
    for (i in 1:SI) {
      lambda0[, t] <- rnorm(SI, mean=param_llm_lambda0[1], sd=param_llm_lambda0[2])
      lambda1_mean_denom <- sum(matriz_Q[i,])
      lambda1_mean <- param_llm_lambda1[1]/lambda1_mean_denom
      lambda1_sd <- param_llm_lambda1[2]
      lambda1[i,,t] <- rnorm(n=4, mean=lambda1_mean, sd=lambda1_sd)
    }
  }
  list(lambda0_llm=lambda0, lambda1_llm=lambda1)
}

### Generación de Lambdas para modelo sLong-LLM
lambda_llm <- f.lambdas_llm(param_llm_lambda0, param_llm_lambda1)
lambda0_llm <- lambda_llm$lambda0_llm
lambda1_llm <- lambda_llm$lambda1_llm

## Calculo de p_nit: probabilidad de acertar respuestas de los items preguntados
### para Hi-Long-DINA
f.p_nit_dina <- function(lambda0, lambda1, alpha){
  alpha_nkt_qikt <- array(NA, dim = c(N, K, T))
  p_nit <- array(NA, dim = c(N, SI, T))
  for (n in 1:N) {
    for (t in 1:T){
      for (i in 1:SI){
        for (k in 1:K){
          alpha_nkt_qikt[n,k,t] <- alpha[n,k,t]^matriz_Q[i,k]
          p_nit[n,i,t] <- exp(lambda0[i,t]+(lambda1[i,t]*prod(alpha_nkt_qikt[n,,t])))
          / (1+exp(lambda0[i,t]+(lambda1[i,t]*prod(alpha_nkt_qikt[n,,t]))))
        }
      }
```

```
    }
  }
  p_nit
}
p_nit_dina <- f.p_nit_dina(lambda0_dina, lambda1_dina, alpha) # obtención p_nit de dina

### para Hi-Long-LLM
f.p_nit_llm <- function(lambda0, lambda1, alpha){
  lambda_alpha_q_temp <- as.data.frame(matrix(NA, nrow=1, ncol=K))
  p_nit <- array(NA, dim = c(N, SI, T))
  for (n in 1:N) {
    for (t in 1:T){
      for (i in 1:SI){
        for (k in 1:K){
          lambda_alpha_q_temp[k] <- lambda1_llm[i,k,t]*alpha[n,k,t]
          *as.numeric(matriz_Q[i,k])
        }
        p_nit[n,i,t]<- exp(lambda0[i,t]+sum(lambda_alpha_q_temp[1,]))/
          (1+exp(lambda0[i,t]+sum(lambda_alpha_q_temp[1,])))
      }
    }
  }
  p_nit
}
p_nit_llm <- f.p_nit_llm(lambda0_llm, lambda1_llm, alpha) # obtención p_nit de llm

### calcular y_nit: Respuestas de personas para las 3 evaluaciones
y_nit_dina <- y_nit_llm <- array(NA, dim = c(N, SI, T))

for (n in 1:N) {
  for (t in 1:T){
    for (i in 1:SI){
      y_nit_dina[n,i,t] <- rbern(1, p_nit_dina[n,i,t])
      y_nit_llm[n,i,t] <- rbern(1, p_nit_llm[n,i,t])
```

```
    }
  }
}

## 4. Consolidado de Respuestas: correctas (1) o fallidas (0)
Respuestas_Y1_dina <- y_nit_dina[,,1]; colnames(Respuestas_Y1_dina) <- c(1:30)
Respuestas_Y2_dina <- y_nit_dina[,,2]; colnames(Respuestas_Y2_dina) <- c(1:30)
Respuestas_Y3_dina <- y_nit_dina[,,3]; colnames(Respuestas_Y3_dina) <- c(1:30)

Respuestas_Y1_llm <- y_nit_llm[,,1]; colnames(Respuestas_Y1_llm) <- c(1:30)
Respuestas_Y2_llm <- y_nit_llm[,,2]; colnames(Respuestas_Y2_llm) <- c(1:30)
Respuestas_Y3_llm <- y_nit_llm[,,3]; colnames(Respuestas_Y3_llm) <- c(1:30)

########## Fin de parte 1 ##########

##### Inicio de parte II: Ejecucición de Simulación por MCMC vía JAGS #####

# 1. Declaración de variables para JAGS
n.simu <- 15000 # numero simulaciones totales para JAGS
n.thin <- 5
n.iter <- n.simu
n.burn <- 7500 # burn in
n.chains <- 2
Q1 <- Q2 <- Q3 <- matriz_Q #para simulacion, matriz Q es igual en todos los t
all.patterns <- atributos_permisibles
jags.inits <- NULL

### Inicio de replicación ###

## Carga de datos
if(mm==1){
  Y1 <- Respuestas_Y1_dina
  Y2 <- Respuestas_Y2_dina
  Y3 <- Respuestas_Y3_dina
```

```
}

if(mm==2){
  Y1 <- Respuestas_Y1_llm
  Y2 <- Respuestas_Y2_llm
  Y3 <- Respuestas_Y3_llm
}

## Generación de lista con inputs del Jags
jags.data <- list("N", "SI", "K", "T", "C", "Y1", "Y2", "Y3", "Q1", "Q2", "Q3",
                  "all.patterns")
model_parameters <- c("g_1","g_2","g_3","s_1","s_2","s_3","alpha","theta")
#parametros a estimar

## Revisión de parámetros jags
print(paste("Modelo de Medición:", mm,"=",mod_medicion,"|" , es_j,"=", est_jerarq))
print(paste("Simulaciones:", n.iter, "||", "Burn in:", n.burn))

## Carga de modelo en JAGS: cosiderando modelo de medición y estructura jerárquica
## Nota3: Copiar código Jags correspondiente para mm y es_j que se usarán
modelo_jags <- function(){
  for(n in 1:N){
    for(i in 1:SI){
      for(k in 1:K){
        w[n,i,k,1] <- alpha[n, k, 1]* Q1[i, k]
        w[n,i,k,2] <- alpha[n, k, 2]* Q2[i, k]
        w[n,i,k,3] <- alpha[n, k, 3]* Q3[i, k]
      }
      eta[n,i,1]<-inprod(lamdaK_1[i,1:K],w[n,i,1:K,1])
      eta[n,i,2]<-inprod(lamdaK_2[i,1:K],w[n,i,1:K,2])
      eta[n,i,3]<-inprod(lamdaK_3[i,1:K],w[n,i,1:K,3])
      logit(p[n,i,1])<-lamda0_1[i]+eta[n,i,1]
      logit(p[n,i,2])<-lamda0_2[i]+eta[n,i,2]
      logit(p[n,i,3])<-lamda0_3[i]+eta[n,i,3]
```

```
    Y1[n, i] ~ dbern(p[n, i, 1])
    Y2[n, i] ~ dbern(p[n, i, 2])
    Y3[n, i] ~ dbern(p[n, i, 3])
  }
  for(t in 1:T){
    for(k in 1:K){
      alpha[n,k,t]<-all.patterns[x[n,t],k]}
    x[n,t]~dcat(pai[n,1:C,t])}}

for(t in 1:T){
  for(n in 1:N){
    for(k in 1:K){logit(m[n, k, t]) <- xi[k] * theta[n,t] - beta[k]}
    pai[n,1,t] <- (1-m[n,1,t])*(1-m[n,1,t])*(1-m[n,3,t])*1
    pai[n,2,t] <- m[n,1,t]*(1-m[n,2,t])*(1-m[n,3,t])*1
    pai[n,3,t] <- (1-m[n,1,t])*m[n,2,t]*(1-m[n,3,t])*1
    pai[n,4,t] <- (1-m[n,1,t])*(1-m[n,2,t])*m[n,3,t]*1
    pai[n,5,t] <- m[n,1,t]*m[n,2,t]*(1-m[n,3,t])*1
    pai[n,6,t] <- m[n,1,t]*(1-m[n,2,t])*m[n,3,t]*1
    pai[n,7,t] <- (1-m[n,1,t])*m[n,2,t]*m[n,3,t]*1
    pai[n,8,t] <- m[n,1,t]*m[n,2,t]*m[n,3,t]*(1-m[n,4,t])
    pai[n,9,t] <- m[n,1,t]*m[n,2,t]*m[n,3,t]*m[n,4,t]
  }}

for(n in 1:N){theta[n,1:T] ~ dmnorm(mu_theta[1:T], pr_theta[1:T, 1:T])}
for(k in 1:K){
  beta[k] ~ dnorm(0, 0.25)
  xi[k] ~ dnorm(0, 0.25) %_% T(0, )}

for(i in 1:SI){
  lamda0_1[i]~dnorm(-2.197,0.25)
  lamda0_2[i]~dnorm(-2.197,0.25)
  lamda0_3[i]~dnorm(-2.197,0.25)
  for(k in 1:K){
    lamdaK_1[i,k]<-xlamdaK_1[i,k]*Q1[i,k]
```

```
      xlamdaK_1[i,k]~dnorm(1,0.25) %_% T(0,)
      lamdaK_2[i,k]<-xlamdaK_2[i,k]*Q2[i,k]
      xlamdaK_2[i,k]~dnorm(1,0.25) %_% T(0,)
      lamdaK_3[i,k]<-xlamdaK_3[i,k]*Q3[i,k]
      xlamdaK_3[i,k]~dnorm(1,0.25) %_% T(0,)
    }
    logit(g_1[i])<-lamda0_1[i]
    logit(ss_1[i])<-lamda0_1[i]+sum(lamdaK_1[i,1:K])
    s_1[i]<-1-ss_1[i]
    logit(g_2[i])<-lamda0_2[i]
    logit(ss_2[i])<-lamda0_2[i]+sum(lamdaK_2[i,1:K])
    s_2[i]<-1-ss_2[i]
    logit(g_3[i])<-lamda0_3[i]
    logit(ss_3[i])<-lamda0_3[i]+sum(lamdaK_3[i,1:K])
    s_3[i]<-1-ss_3[i]
  }
  mu_theta[1] <- 0
  for (t in 2:T){mu_theta[t] ~ dnorm(0, 1)}
  L_theta[1, 1] <- 1
  for(tt in 2:T){
    L_theta[tt, tt] ~ dgamma(1, 1)
    for(ttt in 1:(tt-1)){
      L_theta[tt, ttt] ~ dnorm(0, 1)
      L_theta[ttt, tt] <- 0}}
  Sigma_theta <- L_theta %*% t(L_theta)
  pr_theta[1:T, 1:T] <- inverse(Sigma_theta[1:T, 1:T])
}

# 2. Ejecución de JAGS

print(paste("Inicio de JAGS:", format(Sys.time(),"%Y-%m-%d %X")))
M1 <- jags(data = jags.data, inits = jags.inits,
           parameters.to.save = model_parameters, model.file = modelo_jags,
           n.chains = n.chains, n.iter = n.iter, n.burnin=n.burn, n.thin = 1,
```

```
           DIC = TRUE)
print(paste("Fin de JAGS:", format(Sys.time(),"%Y-%m-%d %X")))

# Revision de rhat para convergencia
sim_rhat <- M1$BUGSoutput$summary[,8] #extracción de rhat para analizar convergencia
rhat_sim_1.2 <- length(sim_rhat[sim_rhat>1.2])
rhat_sim_1.1 <- length(sim_rhat[sim_rhat>1.1])

R_convergence <- sum(M1$BUGSoutput$summary[,8] >= 1.1) == 0
print(paste0("Rhat>1.1: ",rhat_sim_1.1," ||| Rhat>1.2: ",rhat_sim_1.2,
             " ||| Convergencia: ",R_convergence," ||| DIC: ",
             M1$BUGSoutput$DIC))

# ## Autojags si no hay convergencia (opcional)
# ## Nota4: usando autojags, hay veces que mejora y otras que empeora la convergencia
# if(R_convergence == 0){
#   print("Comienza Autojags")
#   update1 <- autojags(M1, n.iter=5000, n.thin=1, n.burnin=2500, Rhat = 1.1,
#                       n.update = 2)
#   print(paste0("Rhat>1.1: ",length(update1[update1$BUGSoutput$summary[,8]>1.1]),
#                " ||| Rhat>1.2: ",length(update1[update1$BUGSoutput$summary[,8]>1.2]),
#                " ||| DIC: ", update1$BUGSoutput$DIC))
#   print(paste(update1$n.iter, "|||", format(Sys.time(),"%Y-%m-%d %X")))
# }

### Fin de ejecución de JAGS ###

########## Fin de parte 2 ##########

#################### Fin de Simulación ####################
```

7.2. Anexo B: Script en R con códigos JAGS para Estudio de Simulación

En este anexo, se presentan los códigos JAGS para cada modelo propuesto usando cada estructura jerárquica secuencial. Para ejecutarlo en el Script de Simulación del Anexo A, solo se debe cambiar el código JAGS que haya en ese script por el que se encuentre en este.

```
############################################################
## Código JAGS para Hi-Long-DINA Lineal
# cuando mm=1 y es_j=1
modelo_jags <- function(){
  for(n in 1:N){
    for(i in 1:SI){
      for(k in 1:K){
        w[n,i,k,1] <- pow(alpha[n, k, 1], Q1[i, k])
        w[n,i,k,2] <- pow(alpha[n, k, 2], Q2[i, k])
        w[n,i,k,3] <- pow(alpha[n, k, 3], Q3[i, k])
      }
      eta[n, i, 1] <- prod(w[n, i, 1:K, 1])
      eta[n, i, 2] <- prod(w[n, i, 1:K, 2])
      eta[n, i, 3] <- prod(w[n, i, 1:K, 3])
      logit(p[n,i,1])<-lamda0_1[i]+lamdaK_1[i]*eta[n,i,1]
      logit(p[n,i,2])<-lamda0_2[i]+lamdaK_2[i]*eta[n,i,2]
      logit(p[n,i,3])<-lamda0_3[i]+lamdaK_3[i]*eta[n,i,3]
      Y1[n, i] ~ dbern(p[n, i, 1])
      Y2[n, i] ~ dbern(p[n, i, 2])
      Y3[n, i] ~ dbern(p[n, i, 3])
    }
    for(t in 1:T){
      for(k in 1:K){
        alpha[n,k,t]<-all.patterns[x[n,t],k]}
      x[n,t]~dcat(pai[n,1:C,t])}}

  for(t in 1:T){
    for(n in 1:N){
```

```
    for(k in 1:K){logit(m[n, k, t]) <- xi[k] * theta[n,t] - beta[k]}
    pai[n,1,t]<-(1-m[n,1,t])
    pai[n,2,t]<-m[n,1,t]*(1-m[n,2,t])
    pai[n,3,t]<-m[n,1,t]*m[n,2,t]*(1-m[n,3,t])
    pai[n,4,t]<-m[n,1,t]*m[n,2,t]*m[n,3,t]*(1-m[n,4,t])
    pai[n,5,t]<-m[n,1,t]*m[n,2,t]*m[n,3,t]*m[n,4,t]
  }}

for(n in 1:N){theta[n,1:T] ~ dmnorm(mu_theta[1:T], pr_theta[1:T, 1:T])}
for(k in 1:K){
  beta[k] ~ dnorm(0, 0.25)
  xi[k] ~ dnorm(0, 0.25) %_% T(0, )}

for(i in 1:SI){
  lamda0_1[i]~dnorm(-2.197,0.25)
  lamda0_2[i]~dnorm(-2.197,0.25)
  lamda0_3[i]~dnorm(-2.197,0.25)
  lamdaK_1[i]~dnorm(4.394,0.25) %_% T(0,)
  lamdaK_2[i]~dnorm(4.394,0.25) %_% T(0,)
  lamdaK_3[i]~dnorm(4.394,0.25) %_% T(0,)
  logit(g_1[i])<-lamda0_1[i]
  logit(ss_1[i])<-lamda0_1[i]+lamdaK_1[i]
  s_1[i]<-1-ss_1[i]
  logit(g_2[i])<-lamda0_2[i]
  logit(ss_2[i])<-lamda0_2[i]+lamdaK_2[i]
  s_2[i]<-1-ss_2[i]
  logit(g_3[i])<-lamda0_3[i]
  logit(ss_3[i])<-lamda0_3[i]+lamdaK_3[i]
  s_3[i]<-1-ss_3[i]
}

mu_theta[1] <- 0
for (t in 2:T){mu_theta[t] ~ dnorm(0, 1)}
L_theta[1, 1] <- 1
```

```
  for(tt in 2:T){
    L_theta[tt, tt] ~ dgamma(1, 1)
    for(ttt in 1:(tt-1)){
      L_theta[tt, ttt] ~ dnorm(0, 1)
      L_theta[ttt, tt] <- 0}}
  Sigma_theta <- L_theta %*% t(L_theta)
  pr_theta[1:T, 1:T] <- inverse(Sigma_theta[1:T, 1:T])
}

############################################################
## Código JAGS para Hi-Long-DINA Convergente
# cuando mm=1 y es_j=2
modelo_jags <- function(){
  for(n in 1:N){
    for(i in 1:SI){
      for(k in 1:K){
        w[n,i,k,1] <- pow(alpha[n, k, 1], Q1[i, k])
        w[n,i,k,2] <- pow(alpha[n, k, 2], Q2[i, k])
        w[n,i,k,3] <- pow(alpha[n, k, 3], Q3[i, k])
      }
      eta[n, i, 1] <- prod(w[n, i, 1:K, 1])
      eta[n, i, 2] <- prod(w[n, i, 1:K, 2])
      eta[n, i, 3] <- prod(w[n, i, 1:K, 3])
      logit(p[n,i,1])<-lamda0_1[i]+lamdaK_1[i]*eta[n,i,1]
      logit(p[n,i,2])<-lamda0_2[i]+lamdaK_2[i]*eta[n,i,2]
      logit(p[n,i,3])<-lamda0_3[i]+lamdaK_3[i]*eta[n,i,3]
      Y1[n, i] ~ dbern(p[n, i, 1])
      Y2[n, i] ~ dbern(p[n, i, 2])
      Y3[n, i] ~ dbern(p[n, i, 3])
    }
    for(t in 1:T){
      for(k in 1:K){
        alpha[n,k,t]<-all.patterns[x[n,t],k]}
      x[n,t]~dcat(pai[n,1:C,t])}}
```

```
for(t in 1:T){
  for(n in 1:N){
    for(k in 1:K){logit(m[n, k, t]) <- xi[k] * theta[n,t] - beta[k]}
    pai[n,1,t] <- (1-m[n,1,t])*(1-m[n,1,t])*(1-m[n,3,t])*1
    pai[n,2,t] <- m[n,1,t]*(1-m[n,2,t])*(1-m[n,3,t])*1
    pai[n,3,t] <- (1-m[n,1,t])*m[n,2,t]*(1-m[n,3,t])*1
    pai[n,4,t] <- (1-m[n,1,t])*(1-m[n,2,t])*m[n,3,t]*1
    pai[n,5,t] <- m[n,1,t]*m[n,2,t]*(1-m[n,3,t])*1
    pai[n,6,t] <- m[n,1,t]*(1-m[n,2,t])*m[n,3,t]*1
    pai[n,7,t] <- (1-m[n,1,t])*m[n,2,t]*m[n,3,t]*1
    pai[n,8,t] <- m[n,1,t]*m[n,2,t]*m[n,3,t]*(1-m[n,4,t])
    pai[n,9,t] <- m[n,1,t]*m[n,2,t]*m[n,3,t]*m[n,4,t]
  }}

for(n in 1:N){theta[n,1:T] ~ dmnorm(mu_theta[1:T], pr_theta[1:T, 1:T])}
for(k in 1:K){
  beta[k] ~ dnorm(0, 0.25)
  xi[k] ~ dnorm(0, 0.25) %_% T(0, )}

for(i in 1:SI){
  lamda0_1[i]~dnorm(-2.197,0.25)
  lamda0_2[i]~dnorm(-2.197,0.25)
  lamda0_3[i]~dnorm(-2.197,0.25)
  lamdaK_1[i]~dnorm(4.394,0.25) %_% T(0,)
  lamdaK_2[i]~dnorm(4.394,0.25) %_% T(0,)
  lamdaK_3[i]~dnorm(4.394,0.25) %_% T(0,)
  logit(g_1[i])<-lamda0_1[i]
  logit(ss_1[i])<-lamda0_1[i]+lamdaK_1[i]
  s_1[i]<-1-ss_1[i]
  logit(g_2[i])<-lamda0_2[i]
  logit(ss_2[i])<-lamda0_2[i]+lamdaK_2[i]
  s_2[i]<-1-ss_2[i]
  logit(g_3[i])<-lamda0_3[i]
```

```
    logit(ss_3[i])<-lamda0_3[i]+lamdaK_3[i]
    s_3[i]<-1-ss_3[i]
  }

  mu_theta[1] <- 0
  for (t in 2:T){mu_theta[t] ~ dnorm(0, 1)}
  L_theta[1, 1] <- 1
  for(tt in 2:T){
    L_theta[tt, tt] ~ dgamma(1, 1)
    for(ttt in 1:(tt-1)){
      L_theta[tt, ttt] ~ dnorm(0, 1)
      L_theta[ttt, tt] <- 0}}
  Sigma_theta <- L_theta %*% t(L_theta)
  pr_theta[1:T, 1:T] <- inverse(Sigma_theta[1:T, 1:T])
}

############################################################
## Código JAGS para Hi-Long-DINA Divergente
# cuando mm=1 y es_j=3
modelo_jags <- function(){
  for(n in 1:N){
    for(i in 1:SI){
      for(k in 1:K){
        w[n,i,k,1] <- pow(alpha[n, k, 1], Q1[i, k])
        w[n,i,k,2] <- pow(alpha[n, k, 2], Q2[i, k])
        w[n,i,k,3] <- pow(alpha[n, k, 3], Q3[i, k])
      }
      eta[n, i, 1] <- prod(w[n, i, 1:K, 1])
      eta[n, i, 2] <- prod(w[n, i, 1:K, 2])
      eta[n, i, 3] <- prod(w[n, i, 1:K, 3])
      logit(p[n,i,1])<-lamda0_1[i]+lamdaK_1[i]*eta[n,i,1]
      logit(p[n,i,2])<-lamda0_2[i]+lamdaK_2[i]*eta[n,i,2]
      logit(p[n,i,3])<-lamda0_3[i]+lamdaK_3[i]*eta[n,i,3]
      Y1[n, i] ~ dbern(p[n, i, 1])
```

```
    Y2[n, i] ~ dbern(p[n, i, 2])
    Y3[n, i] ~ dbern(p[n, i, 3])
  }
  for(t in 1:T){
    for(k in 1:K){
      alpha[n,k,t]<-all.patterns[x[n,t],k]}
    x[n,t]~dcat(pai[n,1:C,t])}}

for(t in 1:T){
  for(n in 1:N){
    for(k in 1:K){logit(m[n, k, t]) <- xi[k] * theta[n,t] - beta[k]}
    pai[n,1,t] <- (1-m[n,1,t])*1*1*1
    pai[n,2,t] <- m[n,1,t]*(1-m[n,2,t])*(1-m[n,3,t])*(1-m[n,4,t])
    pai[n,3,t] <- m[n,1,t]*m[n,2,t]*(1-m[n,3,t])*(1-m[n,4,t])
    pai[n,4,t] <- m[n,1,t]*(1-m[n,2,t])*m[n,3,t]*(1-m[n,4,t])
    pai[n,5,t] <- m[n,1,t]*(1-m[n,2,t])*(1-m[n,3,t])*m[n,4,t]
    pai[n,6,t] <- m[n,1,t]*m[n,2,t]*m[n,3,t]*(1-m[n,4,t])
    pai[n,7,t] <- m[n,1,t]*m[n,2,t]*(1-m[n,3,t])*m[n,4,t]
    pai[n,8,t] <- m[n,1,t]*(1-m[n,2,t])*m[n,3,t]*m[n,4,t]
    pai[n,9,t] <- m[n,1,t]*m[n,2,t]*m[n,3,t]*m[n,4,t]
  }}

for(n in 1:N){theta[n,1:T] ~ dmnorm(mu_theta[1:T], pr_theta[1:T, 1:T])}
for(k in 1:K){
  beta[k] ~ dnorm(0, 0.25)
  xi[k] ~ dnorm(0, 0.25) %_% T(0, )}

for(i in 1:SI){
  lamda0_1[i]~dnorm(-2.197,0.25)
  lamda0_2[i]~dnorm(-2.197,0.25)
  lamda0_3[i]~dnorm(-2.197,0.25)
  lamdaK_1[i]~dnorm(4.394,0.25) %_% T(0,)
  lamdaK_2[i]~dnorm(4.394,0.25) %_% T(0,)
  lamdaK_3[i]~dnorm(4.394,0.25) %_% T(0,)
```

```
    logit(g_1[i])<-lamda0_1[i]
    logit(ss_1[i])<-lamda0_1[i]+lamdaK_1[i]
    s_1[i]<-1-ss_1[i]
    logit(g_2[i])<-lamda0_2[i]
    logit(ss_2[i])<-lamda0_2[i]+lamdaK_2[i]
    s_2[i]<-1-ss_2[i]
    logit(g_3[i])<-lamda0_3[i]
    logit(ss_3[i])<-lamda0_3[i]+lamdaK_3[i]
    s_3[i]<-1-ss_3[i]
  }

  mu_theta[1] <- 0
  for (t in 2:T){mu_theta[t] ~ dnorm(0, 1)}
  L_theta[1, 1] <- 1
  for(tt in 2:T){
    L_theta[tt, tt] ~ dgamma(1, 1)
    for(ttt in 1:(tt-1)){
      L_theta[tt, ttt] ~ dnorm(0, 1)
      L_theta[ttt, tt] <- 0}}
  Sigma_theta <- L_theta %*% t(L_theta)
  pr_theta[1:T, 1:T] <- inverse(Sigma_theta[1:T, 1:T])
}

############################################################
## Código JAGS para Hi-Long-LLM Lineal
# cuando mm=2 y es_j=1
modelo_jags <- function(){
  for(n in 1:N){
    for(i in 1:SI){
      for(k in 1:K){
        w[n,i,k,1] <- alpha[n, k, 1]* Q1[i, k]
        w[n,i,k,2] <- alpha[n, k, 2]* Q2[i, k]
        w[n,i,k,3] <- alpha[n, k, 3]* Q3[i, k]
      }
```

```
    eta[n,i,1]<-inprod(lamdaK_1[i,1:K],w[n,i,1:K,1])
    eta[n,i,2]<-inprod(lamdaK_2[i,1:K],w[n,i,1:K,2])
    eta[n,i,3]<-inprod(lamdaK_3[i,1:K],w[n,i,1:K,3])
    logit(p[n,i,1])<-lamda0_1[i]+eta[n,i,1]
    logit(p[n,i,2])<-lamda0_2[i]+eta[n,i,2]
    logit(p[n,i,3])<-lamda0_3[i]+eta[n,i,3]
    Y1[n, i] ~ dbern(p[n, i, 1])
    Y2[n, i] ~ dbern(p[n, i, 2])
    Y3[n, i] ~ dbern(p[n, i, 3])
  }
  for(t in 1:T){
    for(k in 1:K){
      alpha[n,k,t]<-all.patterns[x[n,t],k]}
    x[n,t]~dcat(pai[n,1:C,t])}}

for(t in 1:T){
  for(n in 1:N){
    for(k in 1:K){logit(m[n, k, t]) <- xi[k] * theta[n,t] - beta[k]}
    pai[n,1,t]<-(1-m[n,1,t])
    pai[n,2,t]<-m[n,1,t]*(1-m[n,2,t])
    pai[n,3,t]<-m[n,1,t]*m[n,2,t]*(1-m[n,3,t])
    pai[n,4,t]<-m[n,1,t]*m[n,2,t]*m[n,3,t]*(1-m[n,4,t])
    pai[n,5,t]<-m[n,1,t]*m[n,2,t]*m[n,3,t]*m[n,4,t]
  }}

for(n in 1:N){theta[n,1:T] ~ dmnorm(mu_theta[1:T], pr_theta[1:T, 1:T])}
for(k in 1:K){
  beta[k] ~ dnorm(0, 0.25)
  xi[k] ~ dnorm(0, 0.25) %_% T(0, )}

for(i in 1:SI){
  lamda0_1[i]~dnorm(-2.197,0.25)
  lamda0_2[i]~dnorm(-2.197,0.25)
  lamda0_3[i]~dnorm(-2.197,0.25)
```

```
    for(k in 1:K){
      lamdaK_1[i,k]<-xlamdaK_1[i,k]*Q1[i,k]
      xlamdaK_1[i,k]~dnorm(1,0.25) %_% T(0,)
      lamdaK_2[i,k]<-xlamdaK_2[i,k]*Q2[i,k]
      xlamdaK_2[i,k]~dnorm(1,0.25) %_% T(0,)
      lamdaK_3[i,k]<-xlamdaK_3[i,k]*Q3[i,k]
      xlamdaK_3[i,k]~dnorm(1,0.25) %_% T(0,)
    }
    logit(g_1[i])<-lamda0_1[i]
    logit(ss_1[i])<-lamda0_1[i]+sum(lamdaK_1[i,1:K])
    s_1[i]<-1-ss_1[i]
    logit(g_2[i])<-lamda0_2[i]
    logit(ss_2[i])<-lamda0_2[i]+sum(lamdaK_2[i,1:K])
    s_2[i]<-1-ss_2[i]
    logit(g_3[i])<-lamda0_3[i]
    logit(ss_3[i])<-lamda0_3[i]+sum(lamdaK_3[i,1:K])
    s_3[i]<-1-ss_3[i]
  }
  mu_theta[1] <- 0
  for (t in 2:T){mu_theta[t] ~ dnorm(0, 1)}
  L_theta[1, 1] <- 1
  for(tt in 2:T){
    L_theta[tt, tt] ~ dgamma(1, 1)
    for(ttt in 1:(tt-1)){
      L_theta[tt, ttt] ~ dnorm(0, 1)
      L_theta[ttt, tt] <- 0}}
  Sigma_theta <- L_theta %*% t(L_theta)
  pr_theta[1:T, 1:T] <- inverse(Sigma_theta[1:T, 1:T])
}

############################################################
## Código JAGS para Hi-Long-LLM Convergente
# cuando mm=2 y es_j=2
modelo_jags <- function(){
```

```
for(n in 1:N){
  for(i in 1:SI){
    for(k in 1:K){
      w[n,i,k,1] <- alpha[n, k, 1]* Q1[i, k]
      w[n,i,k,2] <- alpha[n, k, 2]* Q2[i, k]
      w[n,i,k,3] <- alpha[n, k, 3]* Q3[i, k]
    }
    eta[n,i,1]<-inprod(lamdaK_1[i,1:K],w[n,i,1:K,1])
    eta[n,i,2]<-inprod(lamdaK_2[i,1:K],w[n,i,1:K,2])
    eta[n,i,3]<-inprod(lamdaK_3[i,1:K],w[n,i,1:K,3])
    logit(p[n,i,1])<-lamda0_1[i]+eta[n,i,1]
    logit(p[n,i,2])<-lamda0_2[i]+eta[n,i,2]
    logit(p[n,i,3])<-lamda0_3[i]+eta[n,i,3]
    Y1[n, i] ~ dbern(p[n, i, 1])
    Y2[n, i] ~ dbern(p[n, i, 2])
    Y3[n, i] ~ dbern(p[n, i, 3])
  }
  for(t in 1:T){
    for(k in 1:K){
      alpha[n,k,t]<-all.patterns[x[n,t],k]}
    x[n,t]~dcat(pai[n,1:C,t])}}

for(t in 1:T){
  for(n in 1:N){
    for(k in 1:K){logit(m[n, k, t]) <- xi[k] * theta[n,t] - beta[k]}
    pai[n,1,t] <- (1-m[n,1,t])*(1-m[n,1,t])*(1-m[n,3,t])*1
    pai[n,2,t] <- m[n,1,t]*(1-m[n,2,t])*(1-m[n,3,t])*1
    pai[n,3,t] <- (1-m[n,1,t])*m[n,2,t]*(1-m[n,3,t])*1
    pai[n,4,t] <- (1-m[n,1,t])*(1-m[n,2,t])*m[n,3,t]*1
    pai[n,5,t] <- m[n,1,t]*m[n,2,t]*(1-m[n,3,t])*1
    pai[n,6,t] <- m[n,1,t]*(1-m[n,2,t])*m[n,3,t]*1
    pai[n,7,t] <- (1-m[n,1,t])*m[n,2,t]*m[n,3,t]*1
    pai[n,8,t] <- m[n,1,t]*m[n,2,t]*m[n,3,t]*(1-m[n,4,t])
    pai[n,9,t] <- m[n,1,t]*m[n,2,t]*m[n,3,t]*m[n,4,t]
```

```
  }}

for(n in 1:N){theta[n,1:T] ~ dmnorm(mu_theta[1:T], pr_theta[1:T, 1:T])}
for(k in 1:K){
  beta[k] ~ dnorm(0, 0.25)
  xi[k] ~ dnorm(0, 0.25) %_% T(0, )}

for(i in 1:SI){
  lamda0_1[i]~dnorm(-2.197,0.25)
  lamda0_2[i]~dnorm(-2.197,0.25)
  lamda0_3[i]~dnorm(-2.197,0.25)
  for(k in 1:K){
    lamdaK_1[i,k]<-xlamdaK_1[i,k]*Q1[i,k]
    xlamdaK_1[i,k]~dnorm(1,0.25) %_% T(0,)
    lamdaK_2[i,k]<-xlamdaK_2[i,k]*Q2[i,k]
    xlamdaK_2[i,k]~dnorm(1,0.25) %_% T(0,)
    lamdaK_3[i,k]<-xlamdaK_3[i,k]*Q3[i,k]
    xlamdaK_3[i,k]~dnorm(1,0.25) %_% T(0,)
  }
  logit(g_1[i])<-lamda0_1[i]
  logit(ss_1[i])<-lamda0_1[i]+sum(lamdaK_1[i,1:K])
  s_1[i]<-1-ss_1[i]
  logit(g_2[i])<-lamda0_2[i]
  logit(ss_2[i])<-lamda0_2[i]+sum(lamdaK_2[i,1:K])
  s_2[i]<-1-ss_2[i]
  logit(g_3[i])<-lamda0_3[i]
  logit(ss_3[i])<-lamda0_3[i]+sum(lamdaK_3[i,1:K])
  s_3[i]<-1-ss_3[i]
}
mu_theta[1] <- 0
for (t in 2:T){mu_theta[t] ~ dnorm(0, 1)}
L_theta[1, 1] <- 1
for(tt in 2:T){
  L_theta[tt, tt] ~ dgamma(1, 1)
```

```
    for(ttt in 1:(tt-1)){
      L_theta[tt, ttt] ~ dnorm(0, 1)
      L_theta[ttt, tt] <- 0}}
  Sigma_theta <- L_theta %*% t(L_theta)
  pr_theta[1:T, 1:T] <- inverse(Sigma_theta[1:T, 1:T])
}

############################################################
## Código JAGS para Hi-Long-LLM Divergente
# cuando mm=2 y es_j=3
modelo_jags <- function(){
  for(n in 1:N){
    for(i in 1:SI){
      for(k in 1:K){
        w[n,i,k,1] <- alpha[n, k, 1]* Q1[i, k]
        w[n,i,k,2] <- alpha[n, k, 2]* Q2[i, k]
        w[n,i,k,3] <- alpha[n, k, 3]* Q3[i, k]
      }
      eta[n,i,1]<-inprod(lamdaK_1[i,1:K],w[n,i,1:K,1])
      eta[n,i,2]<-inprod(lamdaK_2[i,1:K],w[n,i,1:K,2])
      eta[n,i,3]<-inprod(lamdaK_3[i,1:K],w[n,i,1:K,3])
      logit(p[n,i,1])<-lamda0_1[i]+eta[n,i,1]
      logit(p[n,i,2])<-lamda0_2[i]+eta[n,i,2]
      logit(p[n,i,3])<-lamda0_3[i]+eta[n,i,3]
      Y1[n, i] ~ dbern(p[n, i, 1])
      Y2[n, i] ~ dbern(p[n, i, 2])
      Y3[n, i] ~ dbern(p[n, i, 3])
    }
    for(t in 1:T){
      for(k in 1:K){
        alpha[n,k,t]<-all.patterns[x[n,t],k]}
      x[n,t]~dcat(pai[n,1:C,t])}}

  for(t in 1:T){
```

```
  for(n in 1:N){
    for(k in 1:K){logit(m[n, k, t]) <- xi[k] * theta[n,t] - beta[k]}
    pai[n,1,t] <- (1-m[n,1,t])*1*1*1
    pai[n,2,t] <- m[n,1,t]*(1-m[n,2,t])*(1-m[n,3,t])*(1-m[n,4,t])
    pai[n,3,t] <- m[n,1,t]*m[n,2,t]*(1-m[n,3,t])*(1-m[n,4,t])
    pai[n,4,t] <- m[n,1,t]*(1-m[n,2,t])*m[n,3,t]*(1-m[n,4,t])
    pai[n,5,t] <- m[n,1,t]*(1-m[n,2,t])*(1-m[n,3,t])*m[n,4,t]
    pai[n,6,t] <- m[n,1,t]*m[n,2,t]*m[n,3,t]*(1-m[n,4,t])
    pai[n,7,t] <- m[n,1,t]*m[n,2,t]*(1-m[n,3,t])*m[n,4,t]
    pai[n,8,t] <- m[n,1,t]*(1-m[n,2,t])*m[n,3,t]*m[n,4,t]
    pai[n,9,t] <- m[n,1,t]*m[n,2,t]*m[n,3,t]*m[n,4,t]
  }}

for(n in 1:N){theta[n,1:T] ~ dmnorm(mu_theta[1:T], pr_theta[1:T, 1:T])}
for(k in 1:K){
  beta[k] ~ dnorm(0, 0.25)
  xi[k] ~ dnorm(0, 0.25) %_% T(0, )}

for(i in 1:SI){
  lamda0_1[i]~dnorm(-2.197,0.25)
  lamda0_2[i]~dnorm(-2.197,0.25)
  lamda0_3[i]~dnorm(-2.197,0.25)
  for(k in 1:K){
    lamdaK_1[i,k]<-xlamdaK_1[i,k]*Q1[i,k]
    xlamdaK_1[i,k]~dnorm(1,0.25) %_% T(0,)
    lamdaK_2[i,k]<-xlamdaK_2[i,k]*Q2[i,k]
    xlamdaK_2[i,k]~dnorm(1,0.25) %_% T(0,)
    lamdaK_3[i,k]<-xlamdaK_3[i,k]*Q3[i,k]
    xlamdaK_3[i,k]~dnorm(1,0.25) %_% T(0,)
  }
  logit(g_1[i])<-lamda0_1[i]
  logit(ss_1[i])<-lamda0_1[i]+sum(lamdaK_1[i,1:K])
  s_1[i]<-1-ss_1[i]
  logit(g_2[i])<-lamda0_2[i]
```

```
    logit(ss_2[i])<-lamda0_2[i]+sum(lamdaK_2[i,1:K])
    s_2[i]<-1-ss_2[i]
    logit(g_3[i])<-lamda0_3[i]
    logit(ss_3[i])<-lamda0_3[i]+sum(lamdaK_3[i,1:K])
    s_3[i]<-1-ss_3[i]
  }
  mu_theta[1] <- 0
  for (t in 2:T){mu_theta[t] ~ dnorm(0, 1)}
  L_theta[1, 1] <- 1
  for(tt in 2:T){
    L_theta[tt, tt] ~ dgamma(1, 1)
    for(ttt in 1:(tt-1)){
      L_theta[tt, ttt] ~ dnorm(0, 1)
      L_theta[ttt, tt] <- 0}}
  Sigma_theta <- L_theta %*% t(L_theta)
  pr_theta[1:T, 1:T] <- inverse(Sigma_theta[1:T, 1:T])
}
```

7.3. Anexo C. Script en R de modelo Hi-Long-DINA para Aplicación

```
# Ejercicio DE Aplicación - Tesis - Hi-Long-DINA

# Limpiar y fijar carpeta
rm(list = ls())
setwd(dirname(rstudioapi::getActiveDocumentContext()$path))

# Nota1: Descomentar para instalar librerías (con: Ctrl + Shift + c)

# 1. Instalación y Carga de librerias
# suppressMessages(install.packages("R2jags"))
# suppressMessages(install.packages("dplyr"))
# suppressMessages(install.packages("stringr"))
suppressMessages(library(R2jags))
suppressMessages(library(dplyr))
suppressMessages(library(stringr))

# 2. Carga de inputs: Matriz Q, Perfiles Permisibles
#carga de matrices Q y perfiles permisibles para cada estructura secuencial
load("aplicacion_inputs.RData")

#Generación de inputs
Q1 <- Q2 <- Q3 <- matriz_Q #para aplicacion, matriz Q es igual en todos los t

### Carga de datasets con respuestas de estudiantes
Y1 <- as.data.frame(respuestas_individuos[,1:18]);
Y2 <- as.data.frame(respuestas_individuos[,19:36]);
Y3 <- as.data.frame(respuestas_individuos[,37:54]);

### Parámetros de estudio
N = dim(respuestas_individuos)[1] #Tamaño de la muestra / individuos (loop n);
SI = dim(matriz_Q)[1] #Número de ítems en cada punto de tiempo (loop i);
```

```
K = dim(matriz_Q)[2] #Número de atributos (loop k);
T = 3 #Número de puntos de tiempo (loop t);
mm = 1 #modelo de medicion: 1=DINA, 2=LLM
C = dim(all.patterns)[1] # C* = número de perfiles permisibles

## Configurar parámetros de la simulación
n_simu = 15000 #numero simulaciones
n.thin<- 5
n.iter <- n_simu * n.thin
n.burn <- 7500
n.chains <- 2

# Revisión
mod_medicion <- case_when(mm == 1 ~ "HiLongDINA", mm == 2 ~ "HiLongLLM")

print(paste("Parametros: ", "Tamaño de Muestra:",N,"|| N. Items:",SI,"||
            N. Atributos:",K,"|| Tiempos:",T,"|| Modelo de Medición:",
            mod_medicion ,"|| N. Perfiles Permitidos:",C,"|| N. Simulaciones:",
            n_simu))

# 3. Ejecución de Jags

## Carga de modelo en JAGS
hi_long_DINA <- function(){
  for(n in 1:N){
    for(i in 1:SI){
      for(k in 1:K){
        w[n,i,k,1] <- pow(alpha[n, k, 1], Q1[i, k])
        w[n,i,k,2] <- pow(alpha[n, k, 2], Q2[i, k])
        w[n,i,k,3] <- pow(alpha[n, k, 3], Q3[i, k])
      }
      eta[n, i, 1] <- prod(w[n, i, 1:K, 1])
      eta[n, i, 2] <- prod(w[n, i, 1:K, 2])
      eta[n, i, 3] <- prod(w[n, i, 1:K, 3])
```

```
      logit(p[n,i,1])<-lamda0_1[i]+lamdaK_1[i]*eta[n,i,1]
      logit(p[n,i,2])<-lamda0_2[i]+lamdaK_2[i]*eta[n,i,2]
      logit(p[n,i,3])<-lamda0_3[i]+lamdaK_3[i]*eta[n,i,3]
      Y1[n, i] ~ dbern(p[n, i, 1])
      Y2[n, i] ~ dbern(p[n, i, 2])
      Y3[n, i] ~ dbern(p[n, i, 3])
    }
    for(t in 1:T){
      for(k in 1:K){
        alpha[n,k,t]<-all.patterns[x[n,t],k]}
      x[n,t]~dcat(pai[n,1:C,t])}}

  for(t in 1:T){
    for(n in 1:N){
      for(k in 1:K){logit(m[n, k, t]) <- xi[k] * theta[n,t] - beta[k]}
      pai[n,1,t] <- (1-m[n,1,t])*(1-m[n,2,t])*(1-m[n,3,t])*(1-m[n,4,t])*
(1-m[n,5,t])*1
      pai[n,2,t] <- m[n,1,t]*(1-m[n,2,t])*(1-m[n,3,t])*(1-m[n,4,t])*
(1-m[n,5,t])*1
      pai[n,3,t] <- (1-m[n,1,t])*m[n,2,t]*(1-m[n,3,t])*(1-m[n,4,t])*
(1-m[n,5,t])*1
      pai[n,4,t] <- (1-m[n,1,t])*(1-m[n,2,t])*m[n,3,t]*(1-m[n,4,t])*
(1-m[n,5,t])*1
      pai[n,5,t] <- m[n,1,t]*m[n,2,t]*(1-m[n,3,t])*(1-m[n,4,t])*(1-m[n,5,t])*1
      pai[n,6,t] <- m[n,1,t]*(1-m[n,2,t])*m[n,3,t]*(1-m[n,4,t])*(1-m[n,5,t])*1
      pai[n,7,t] <- (1-m[n,1,t])*m[n,2,t]*m[n,3,t]*(1-m[n,4,t])*(1-m[n,5,t])*1
      pai[n,8,t] <- m[n,1,t]*m[n,2,t]*m[n,3,t]*(1-m[n,4,t])*(1-m[n,5,t])*1
      pai[n,9,t] <- (1-m[n,1,t])*(1-m[n,2,t])*(1-m[n,3,t])*m[n,4,t]*(1-m[n,5,t])*1
      pai[n,10,t] <- m[n,1,t]*(1-m[n,2,t])*(1-m[n,3,t])*m[n,4,t]*(1-m[n,5,t])*1
      pai[n,11,t] <- (1-m[n,1,t])*m[n,2,t]*(1-m[n,3,t])*m[n,4,t]*(1-m[n,5,t])*1
      pai[n,12,t] <- (1-m[n,1,t])*(1-m[n,2,t])*m[n,3,t]*m[n,4,t]*(1-m[n,5,t])*1
      pai[n,13,t] <- m[n,1,t]*m[n,2,t]*(1-m[n,3,t])*m[n,4,t]*(1-m[n,5,t])*1
      pai[n,14,t] <- m[n,1,t]*(1-m[n,2,t])*m[n,3,t]*m[n,4,t]*(1-m[n,5,t])*1
      pai[n,15,t] <- (1-m[n,1,t])*m[n,2,t]*m[n,3,t]*m[n,4,t]*(1-m[n,5,t])*1
```

```
      pai[n,16,t] <- m[n,1,t]*m[n,2,t]*m[n,3,t]*m[n,4,t]*(1-m[n,5,t])*1
      pai[n,17,t] <- (1-m[n,1,t])*(1-m[n,2,t])*(1-m[n,3,t])*(1-m[n,4,t])*m[n,5,t]*1
      pai[n,18,t] <- m[n,1,t]*(1-m[n,2,t])*(1-m[n,3,t])*(1-m[n,4,t])*m[n,5,t]*1
      pai[n,19,t] <- (1-m[n,1,t])*m[n,2,t]*(1-m[n,3,t])*(1-m[n,4,t])*m[n,5,t]*1
      pai[n,20,t] <- (1-m[n,1,t])*(1-m[n,2,t])*m[n,3,t]*(1-m[n,4,t])*m[n,5,t]*1
      pai[n,21,t] <- m[n,1,t]*m[n,2,t]*(1-m[n,3,t])*(1-m[n,4,t])*m[n,5,t]*1
      pai[n,22,t] <- m[n,1,t]*(1-m[n,2,t])*m[n,3,t]*(1-m[n,4,t])*m[n,5,t]*1
      pai[n,23,t] <- (1-m[n,1,t])*m[n,2,t]*m[n,3,t]*(1-m[n,4,t])*m[n,5,t]*1
      pai[n,24,t] <- m[n,1,t]*m[n,2,t]*m[n,3,t]*(1-m[n,4,t])*m[n,5,t]*1
      pai[n,25,t] <- (1-m[n,1,t])*(1-m[n,2,t])*(1-m[n,3,t])*m[n,4,t]*
m[n,5,t]*(1-m[n,6,t])
      pai[n,26,t] <- m[n,1,t]*(1-m[n,2,t])*(1-m[n,3,t])*m[n,4,t]*m[n,5,t]*
(1-m[n,6,t])
      pai[n,27,t] <- (1-m[n,1,t])*m[n,2,t]*(1-m[n,3,t])*m[n,4,t]*m[n,5,t]*
(1-m[n,6,t])
      pai[n,28,t] <- (1-m[n,1,t])*(1-m[n,2,t])*m[n,3,t]*m[n,4,t]*m[n,5,t]*
(1-m[n,6,t])
      pai[n,29,t] <- m[n,1,t]*m[n,2,t]*(1-m[n,3,t])*m[n,4,t]*m[n,5,t]*
(1-m[n,6,t])
      pai[n,30,t] <- m[n,1,t]*(1-m[n,2,t])*m[n,3,t]*m[n,4,t]*m[n,5,t]*(1-m[n,6,t])
      pai[n,31,t] <- (1-m[n,1,t])*m[n,2,t]*m[n,3,t]*m[n,4,t]*m[n,5,t]*(1-m[n,6,t])
      pai[n,32,t] <- m[n,1,t]*m[n,2,t]*m[n,3,t]*m[n,4,t]*m[n,5,t]*(1-m[n,6,t])
      pai[n,33,t] <- (1-m[n,1,t])*(1-m[n,2,t])*(1-m[n,3,t])*m[n,4,t]*m[n,5,t]
*m[n,6,t]
      pai[n,34,t] <- m[n,1,t]*(1-m[n,2,t])*(1-m[n,3,t])*m[n,4,t]*m[n,5,t]*
m[n,6,t]
      pai[n,35,t] <- (1-m[n,1,t])*m[n,2,t]*(1-m[n,3,t])*m[n,4,t]*m[n,5,t]*
m[n,6,t]
      pai[n,36,t] <- (1-m[n,1,t])*(1-m[n,2,t])*m[n,3,t]*m[n,4,t]*m[n,5,t]*
m[n,6,t]
      pai[n,37,t] <- m[n,1,t]*m[n,2,t]*(1-m[n,3,t])*m[n,4,t]*m[n,5,t]*m[n,6,t]
      pai[n,38,t] <- m[n,1,t]*(1-m[n,2,t])*m[n,3,t]*m[n,4,t]*m[n,5,t]*m[n,6,t]
      pai[n,39,t] <- (1-m[n,1,t])*m[n,2,t]*m[n,3,t]*m[n,4,t]*m[n,5,t]*m[n,6,t]
      pai[n,40,t] <- m[n,1,t]*m[n,2,t]*m[n,3,t]*m[n,4,t]*m[n,5,t]*m[n,6,t]
```

```
  }}

for(n in 1:N){theta[n,1:T] ~ dmnorm(mu_theta[1:T], pr_theta[1:T, 1:T])}
for(k in 1:K){
  beta[k] ~ dnorm(0, 0.25)
  xi[k] ~ dnorm(0, 0.25) %_% T(0, )}

for(i in 1:SI){
  lamda0_1[i]~dnorm(-2.197,0.25)
  lamda0_2[i]~dnorm(-2.197,0.25)
  lamda0_3[i]~dnorm(-2.197,0.25)
  lamdaK_1[i]~dnorm(4.394,0.25) %_% T(0,)
  lamdaK_2[i]~dnorm(4.394,0.25) %_% T(0,)
  lamdaK_3[i]~dnorm(4.394,0.25) %_% T(0,)
  logit(g_1[i])<-lamda0_1[i]
  logit(ss_1[i])<-lamda0_1[i]+lamdaK_1[i]
  s_1[i]<-1-ss_1[i]
  logit(g_2[i])<-lamda0_2[i]
  logit(ss_2[i])<-lamda0_2[i]+lamdaK_2[i]
  s_2[i]<-1-ss_2[i]
  logit(g_3[i])<-lamda0_3[i]
  logit(ss_3[i])<-lamda0_3[i]+lamdaK_3[i]
  s_3[i]<-1-ss_3[i]
}

mu_theta[1] <- 0
for (t in 2:T){mu_theta[t] ~ dnorm(0, 1)}
L_theta[1, 1] <- 1
for(tt in 2:T){
  L_theta[tt, tt] ~ dgamma(1, 1)
  for(ttt in 1:(tt-1)){
    L_theta[tt, ttt] ~ dnorm(0, 1)
    L_theta[ttt, tt] <- 0}}
Sigma_theta <- L_theta %*% t(L_theta)
```

```
  pr_theta[1:T, 1:T] <- inverse(Sigma_theta[1:T, 1:T])
}

## Generación de lista con inputs del Jags
jags.data <- list("N", "SI", "K", "T", "C", "Y1", "Y2", "Y3", "Q1", "Q2", "Q3",
                  "all.patterns")
model_parameters <- c("g_1","g_2","g_3","s_1","s_2","s_3","p","pai","theta")
jags.inits <- NULL #autor no define inits

##### Ejecución de JAGS #####
set.seed(10000003)
M1 <- jags(data = jags.data, inits = jags.inits, parameters.to.save = model_parameters,
           model.file = Hi_long_LLM, n.chains = n.chains, n.iter = n.iter,
           n.burnin=n.burn, n.thin = n.thin, DIC = TRUE)

mcmc1 = as.mcmc(M1)
gelman.diag(mcmc1)
traceplot(mcmc1[,"g_1"])

# revision de rhat para convergencia
sim_rhat <- M1$BUGSoutput$summary[,8] #extracción de rhat para analizar convergencia
rhat_sim_1.2 <- length(sim_rhat[sim_rhat>1.2])
rhat_sim_1.1 <- length(sim_rhat[sim_rhat>1.1])
R_convergence <- sum(M1$BUGSoutput$summary[,8] >= 1.1) == 0

R_convergence <- sum(M1$BUGSoutput$summary[,8] >= 1.1) == 0
print(paste0("Rhat>1.1: ",rhat_sim_1.1," ||| Rhat>1.2: ",rhat_sim_1.2,
             " ||| Convergencia: ",R_convergence," ||| DIC: ", M1$BUGSoutput$DIC))

## Autojags si no hay convergencia (opcional)
## Nota2: usando autojags, hay veces que mejora y otras que empeora la convergencia
# if(R_convergence == 0){
#   print("Comienza Autojags")
#   update1 <- autojags(M1, n.iter=5000, n.thin=1, n.burnin=2500,
```

```
# Rhat = 1.1, n.update = 2)
#   print(paste0("Rhat>1.1: ",length(update1[update1$BUGSoutput$summary[,8]>1.1]),
#                " ||| Rhat>1.2: ",length(update1[update1$BUGSoutput$summary[,8]>1.2]),
#                " ||| DIC: ", update1$BUGSoutput$DIC))
#   print(paste(update1$n.iter, "|||", format(Sys.time(),"%Y-%m-%d %X")))
# }

### Fin de ejecución de JAGS ###

##################### Fin de script ####################
```

7.4. Anexo D. Script en R de modelo Hi-Long-LLM para Aplicación

```
# Ejercicio DE Aplicación - Tesis - Hi-Long-LLM

# Limpiar y fijar carpeta
rm(list = ls())
setwd(dirname(rstudioapi::getActiveDocumentContext()$path))

# Nota1: Descomentar para instalar librerías (con: Ctrl + Shift + c)
# 1. Instalación y Carga de librerias
# suppressMessages(install.packages("R2jags"))
# suppressMessages(install.packages("dplyr"))
# suppressMessages(install.packages("stringr"))
suppressMessages(library(R2jags))
suppressMessages(library(dplyr))
suppressMessages(library(stringr))

# 2. Carga de inputs: Matriz Q, Perfiles Permisibles
##indicar folder donde se almacenarán los datos
load("aplicacion_inputs.RData") #carga de matrices Q y perfiles permisibles

#Generación de inputs
Q1 <- Q2 <- Q3 <- matriz_Q #para aplicacion, matriz Q es igual en todos los t

### Carga de datasets con respuestas de estudiantes
Y1 <- as.data.frame(respuestas_individuos[,1:18])
Y2 <- as.data.frame(respuestas_individuos[,19:36])
Y3 <- as.data.frame(respuestas_individuos[,37:54])

### Parámetros de estudio
N = dim(respuestas_individuos)[1] #Tamaño de la muestra / individuos (loop n);
SI = dim(matriz_Q)[1] #Número de ítems en cada punto de tiempo (loop i);
K = dim(matriz_Q)[2] #Número de atributos (loop k);
```

```
T = 3 #Número de puntos de tiempo (loop t);
mm = 2 #modelo de medicion: 1=DINA, 2=LLM
C = dim(all.patterns)[1] # C* = número de perfiles permisibles

## Configurar parámetros de la simulación
n_simu = 15000 #numero simulaciones
n.thin <- 2
n.iter <- n_simu * n.thin
n.burn <- 5000
n.chains <- 2

# 18000, 20000
# 5,      10
#
# 8000,  10000
# 2

# Revisión
mod_medicion <- case_when(mm == 1 ~ "HiLongDINA", mm == 2 ~ "HiLongLLM")

print(paste("Parametros: ", "Tamaño de Muestra:",N,"|| N. Items:",SI,
            "|| N. Atributos:",K,"|| Tiempos:",T,"|| Modelo de Medición:",
            mod_medicion ,"|| N. Perfiles Permitidos:",C,"|| N. Simulaciones:",
            n_simu))

# 3. Ejecución de Jags

## Carga de modelo en JAGS
Hi_long_LLM <- function(){
  for(n in 1:N){
    for(i in 1:SI){
      for(k in 1:K){
        w[n,i,k,1] <- alpha[n, k, 1]* Q1[i, k]
        w[n,i,k,2] <- alpha[n, k, 2]* Q2[i, k]
```

```
        w[n,i,k,3] <- alpha[n, k, 3]* Q3[i, k]
      }
      eta[n,i,1]<-inprod(lamdaK_1[i,1:K],w[n,i,1:K,1])
      eta[n,i,2]<-inprod(lamdaK_2[i,1:K],w[n,i,1:K,2])
      eta[n,i,3]<-inprod(lamdaK_3[i,1:K],w[n,i,1:K,3])
      logit(p[n,i,1])<-lamda0_1[i]+eta[n,i,1]
      logit(p[n,i,2])<-lamda0_2[i]+eta[n,i,2]
      logit(p[n,i,3])<-lamda0_3[i]+eta[n,i,3]
      Y1[n, i] ~ dbern(p[n, i, 1])
      Y2[n, i] ~ dbern(p[n, i, 2])
      Y3[n, i] ~ dbern(p[n, i, 3])
    }
    for(t in 1:T){
      for(k in 1:K){
        alpha[n,k,t]<-all.patterns[x[n,t],k]}
      x[n,t]~dcat(pai[n,1:C,t])}}

  for(t in 1:T){
    for(n in 1:N){
      for(k in 1:K){logit(m[n, k, t]) <- xi[k] * theta[n,t] - beta[k]}
      pai[n,1,t] <- (1-m[n,1,t])*(1-m[n,2,t])*(1-m[n,3,t])*(1-m[n,4,t])*
(1-m[n,5,t])*1
      pai[n,2,t] <- m[n,1,t]*(1-m[n,2,t])*(1-m[n,3,t])*(1-m[n,4,t])*
(1-m[n,5,t])*1
      pai[n,3,t] <- (1-m[n,1,t])*m[n,2,t]*(1-m[n,3,t])*(1-m[n,4,t])*
(1-m[n,5,t])*1
      pai[n,4,t] <- (1-m[n,1,t])*(1-m[n,2,t])*m[n,3,t]*(1-m[n,4,t])*
(1-m[n,5,t])*1
      pai[n,5,t] <- m[n,1,t]*m[n,2,t]*(1-m[n,3,t])*(1-m[n,4,t])*(1-m[n,5,t])*1
      pai[n,6,t] <- m[n,1,t]*(1-m[n,2,t])*m[n,3,t]*(1-m[n,4,t])*(1-m[n,5,t])*1
      pai[n,7,t] <- (1-m[n,1,t])*m[n,2,t]*m[n,3,t]*(1-m[n,4,t])*(1-m[n,5,t])*1
      pai[n,8,t] <- m[n,1,t]*m[n,2,t]*m[n,3,t]*(1-m[n,4,t])*(1-m[n,5,t])*1
      pai[n,9,t] <- (1-m[n,1,t])*(1-m[n,2,t])*(1-m[n,3,t])*m[n,4,t]*
(1-m[n,5,t])*1
```

```
      pai[n,10,t] <- m[n,1,t]*(1-m[n,2,t])*(1-m[n,3,t])*m[n,4,t]*(1-m[n,5,t])*1
      pai[n,11,t] <- (1-m[n,1,t])*m[n,2,t]*(1-m[n,3,t])*m[n,4,t]*(1-m[n,5,t])*1
      pai[n,12,t] <- (1-m[n,1,t])*(1-m[n,2,t])*m[n,3,t]*m[n,4,t]*(1-m[n,5,t])*1
      pai[n,13,t] <- m[n,1,t]*m[n,2,t]*(1-m[n,3,t])*m[n,4,t]*(1-m[n,5,t])*1
      pai[n,14,t] <- m[n,1,t]*(1-m[n,2,t])*m[n,3,t]*m[n,4,t]*(1-m[n,5,t])*1
      pai[n,15,t] <- (1-m[n,1,t])*m[n,2,t]*m[n,3,t]*m[n,4,t]*(1-m[n,5,t])*1
      pai[n,16,t] <- m[n,1,t]*m[n,2,t]*m[n,3,t]*m[n,4,t]*(1-m[n,5,t])*1
      pai[n,17,t] <- (1-m[n,1,t])*(1-m[n,2,t])*(1-m[n,3,t])*(1-m[n,4,t])*
m[n,5,t]*1
      pai[n,18,t] <- m[n,1,t]*(1-m[n,2,t])*(1-m[n,3,t])*(1-m[n,4,t])*
m[n,5,t]*1
      pai[n,19,t] <- (1-m[n,1,t])*m[n,2,t]*(1-m[n,3,t])*(1-m[n,4,t])*m[n,5,t]*1
      pai[n,20,t] <- (1-m[n,1,t])*(1-m[n,2,t])*m[n,3,t]*(1-m[n,4,t])*m[n,5,t]*1
      pai[n,21,t] <- m[n,1,t]*m[n,2,t]*(1-m[n,3,t])*(1-m[n,4,t])*m[n,5,t]*1
      pai[n,22,t] <- m[n,1,t]*(1-m[n,2,t])*m[n,3,t]*(1-m[n,4,t])*m[n,5,t]*1
      pai[n,23,t] <- (1-m[n,1,t])*m[n,2,t]*m[n,3,t]*(1-m[n,4,t])*m[n,5,t]*1
      pai[n,24,t] <- m[n,1,t]*m[n,2,t]*m[n,3,t]*(1-m[n,4,t])*m[n,5,t]*1
      pai[n,25,t] <- (1-m[n,1,t])*(1-m[n,2,t])*(1-m[n,3,t])*m[n,4,t]*m[n,5,t]*
(1-m[n,6,t])
      pai[n,26,t] <- m[n,1,t]*(1-m[n,2,t])*(1-m[n,3,t])*m[n,4,t]*m[n,5,t]*
(1-m[n,6,t])
      pai[n,27,t] <- (1-m[n,1,t])*m[n,2,t]*(1-m[n,3,t])*m[n,4,t]*m[n,5,t]*
(1-m[n,6,t])
      pai[n,28,t] <- (1-m[n,1,t])*(1-m[n,2,t])*m[n,3,t]*m[n,4,t]*m[n,5,t]*
(1-m[n,6,t])
      pai[n,29,t] <- m[n,1,t]*m[n,2,t]*(1-m[n,3,t])*m[n,4,t]*m[n,5,t]*
(1-m[n,6,t])
      pai[n,30,t] <- m[n,1,t]*(1-m[n,2,t])*m[n,3,t]*m[n,4,t]*m[n,5,t]*
(1-m[n,6,t])
      pai[n,31,t] <- (1-m[n,1,t])*m[n,2,t]*m[n,3,t]*m[n,4,t]*m[n,5,t]*
(1-m[n,6,t])
      pai[n,32,t] <- m[n,1,t]*m[n,2,t]*m[n,3,t]*m[n,4,t]*m[n,5,t]*(1-m[n,6,t])
      pai[n,33,t] <- (1-m[n,1,t])*(1-m[n,2,t])*(1-m[n,3,t])*m[n,4,t]*
m[n,5,t]*m[n,6,t]
```

```
      pai[n,34,t] <- m[n,1,t]*(1-m[n,2,t])*(1-m[n,3,t])*m[n,4,t]*m[n,5,t]*
m[n,6,t]
      pai[n,35,t] <- (1-m[n,1,t])*m[n,2,t]*(1-m[n,3,t])*m[n,4,t]*m[n,5,t]*
m[n,6,t]
      pai[n,36,t] <- (1-m[n,1,t])*(1-m[n,2,t])*m[n,3,t]*m[n,4,t]*m[n,5,t]*
m[n,6,t]
      pai[n,37,t] <- m[n,1,t]*m[n,2,t]*(1-m[n,3,t])*m[n,4,t]*m[n,5,t]*m[n,6,t]
      pai[n,38,t] <- m[n,1,t]*(1-m[n,2,t])*m[n,3,t]*m[n,4,t]*m[n,5,t]*m[n,6,t]
      pai[n,39,t] <- (1-m[n,1,t])*m[n,2,t]*m[n,3,t]*m[n,4,t]*m[n,5,t]*m[n,6,t]
      pai[n,40,t] <- m[n,1,t]*m[n,2,t]*m[n,3,t]*m[n,4,t]*m[n,5,t]*m[n,6,t]
    }}


  for(n in 1:N){theta[n,1:T] ~ dmnorm(mu_theta[1:T], pr_theta[1:T, 1:T])}
  for(k in 1:K){
    beta[k] ~ dnorm(0, 0.25)
    xi[k] ~ dnorm(0, 0.25) %_% T(0, )}

  for(i in 1:SI){
    lamda0_1[i]~dnorm(-2.197,0.25)
    lamda0_2[i]~dnorm(-2.197,0.25)
    lamda0_3[i]~dnorm(-2.197,0.25)
    for(k in 1:K){
      lamdaK_1[i,k]<-xlamdaK_1[i,k]*Q1[i,k]
      xlamdaK_1[i,k]~dnorm(1,0.25) %_% T(0,)
      lamdaK_2[i,k]<-xlamdaK_2[i,k]*Q2[i,k]
      xlamdaK_2[i,k]~dnorm(1,0.25) %_% T(0,)
      lamdaK_3[i,k]<-xlamdaK_3[i,k]*Q3[i,k]
      xlamdaK_3[i,k]~dnorm(1,0.25) %_% T(0,)
    }
    logit(g_1[i])<-lamda0_1[i]
    logit(ss_1[i])<-lamda0_1[i]+sum(lamdaK_1[i,1:K])
    s_1[i]<-1-ss_1[i]
    logit(g_2[i])<-lamda0_2[i]
```

```
    logit(ss_2[i])<-lamda0_2[i]+sum(lamdaK_2[i,1:K])
    s_2[i]<-1-ss_2[i]
    logit(g_3[i])<-lamda0_3[i]
    logit(ss_3[i])<-lamda0_3[i]+sum(lamdaK_3[i,1:K])
    s_3[i]<-1-ss_3[i]
  }
  mu_theta[1] <- 0
  for (t in 2:T){mu_theta[t] ~ dnorm(0, 1)}
  L_theta[1, 1] <- 1
  for(tt in 2:T){
    L_theta[tt, tt] ~ dgamma(1, 1)
    for(ttt in 1:(tt-1)){
      L_theta[tt, ttt] ~ dnorm(0, 1)
      L_theta[ttt, tt] <- 0}}
  Sigma_theta <- L_theta %*% t(L_theta)
  pr_theta[1:T, 1:T] <- inverse(Sigma_theta[1:T, 1:T])
}

## Generación de lista con inputs del Jags
jags.data <- list("N", "SI", "K", "T", "C", "Y1", "Y2", "Y3", "Q1", "Q2", "Q3",
                  "all.patterns")
model_parameters <- c("g_1","g_2","g_3","s_1","s_2","s_3","p","pai","theta")

#jags.inits <- list(inits1,inits2)

jags.inits <- NULL #autor no define inits

##### Ejecución de JAGS #####
set.seed(10000003)
M1 <- jags(data = jags.data, inits = jags.inits,
           parameters.to.save = model_parameters,
           model.file = Hi_long_LLM, n.chains = n.chains, n.iter = n.iter,
           n.burnin=n.burn, n.thin = n.thin, DIC = TRUE)
```

```
# revision de rhat para convergencia
sim_rhat <- M1$BUGSoutput$summary[,8] #extracción de rhat para analizar convergencia
rhat_sim_1.2 <- length(sim_rhat[sim_rhat>1.2])
rhat_sim_1.1 <- length(sim_rhat[sim_rhat>1.1])
R_convergence <- sum(M1$BUGSoutput$summary[,8] >= 1.1) == 0

R_convergence <- sum(M1$BUGSoutput$summary[,8] >= 1.1) == 0
print(paste0("Rhat>1.1: ",rhat_sim_1.1," ||| Rhat>1.2: ",rhat_sim_1.2,
             " ||| Convergencia: ",R_convergence," ||| DIC: ", M1$BUGSoutput$DIC))

# ## Autojags si no hay convergencia (opcional)
# ## Nota2: usando autojags, hay veces que mejora y otras que empeora la convergencia
# if(R_convergence == 0){
#   print("Comienza Autojags")
#   update1 <- autojags(M1, n.iter=5000, n.thin=1, n.burnin=2500, Rhat = 1.1,
# n.update=2)
#   print(paste0("Rhat>1.1: ",length(update1[update1$BUGSoutput$summary[,8]>1.1]),
#                " ||| Rhat>1.2: ",length(update1[update1$BUGSoutput$summary[,8]>1.2]),
#                " ||| DIC: ", update1$BUGSoutput$DIC))
#   print(paste(update1$n.iter, "|||", format(Sys.time(),"%Y-%m-%d %X")))
# }

### Fin de ejecución de JAGS ###

##################### Fin de script ####################
```

7.5. Anexo E. Proporciones combinadas por perfil en Aplicación

A continuación, se muestran las proporciones combinadas de los perfiles de cada modelo:

Tabla 7.1: Resultados de Proporciones combinadas de modelos secuenciales

		Hi-Long-DINA			Hi-Long-LLM		
Núm.	Perfiles	*t=1*	*t=2*	*t=3*	*t=1*	*t=2*	*t=3*
1	000000	0.107	0.106	0.048	0.086	0.110	0.093
2	100000	0.094	0.091	0.059	0.115	0.095	0.092
3	010000	0.022	0.021	0.014	0.019	0.015	0.015
4	001000	0.001	0.001	0.001	0.001	0	0
5	110000	0.064	0.062	0.056	0.079	0.061	0.062
6	101000	0.004	0.004	0.004	0.003	0.003	0.003
7	011000	0.001	0.001	0.001	0.001	0.001	0.001
8	111000	0.009	0.008	0.012	0.011	0.008	0.009
9	000100	0.006	0.006	0.004	0.005	0.003	0.003
10	100100	0.024	0.024	0.025	0.022	0.017	0.017
11	010100	0.006	0.006	0.007	0.009	0.007	0.007
12	001100	0	0	0.001	0	0	0
13	110100	0.049	0.048	0.068	0.073	0.054	0.059
14	101100	0.006	0.006	0.008	0.005	0.003	0.004
15	011100	0.002	0.002	0.003	0.004	0.003	0.003
16	111100	0.043	0.04	0.051	0.063	0.037	0.041
17	000010	0	0	0	0	0	0
18	100010	0.001	0.001	0.001	0.001	0.001	0.001
19	010010	0	0	0	0	0	0
20	001010	0	0	0	0	0	0
21	110010	0.002	0.002	0.003	0.004	0.003	0.003
22	101010	0	0	0.001	0	0	0
23	011010	0	0	0	0	0	0
24	111010	0.006	0.005	0.005	0.006	0.004	0.004
25	000110	0	0	0	0	0	0
26	100110	0.001	0.001	0.001	0.001	0.001	0.001
27	010110	0	0	0	0.001	0.001	0.001
28	001110	0	0	0	0	0	0
29	110110	0.004	0.004	0.005	0.018	0.010	0.012
30	101110	0.001	0.001	0.001	0.002	0.001	0.001
31	011110	0	0	0	0.004	0.002	0.003
32	111110	0.024	0.023	0.022	0.124	0.093	0.103
33	000111	0	0	0	0	0	0
34	100111	0.002	0.001	0.002	0.001	0	0
35	010111	0.001	0.001	0.001	0.001	0.001	0.001
36	001111	0	0	0	0	0	0
37	110111	0.019	0.018	0.019	0.013	0.008	0.009
38	101111	0.009	0.009	0.009	0.002	0.002	0.002
39	011111	0.004	0.004	0.004	0.006	0.005	0.006
40	111111	0.488	0.504	0.563	0.318	0.451	0.443

Tabla 7.2: Resultados de Proporciones combinadas de modelos no secuenciales (I)

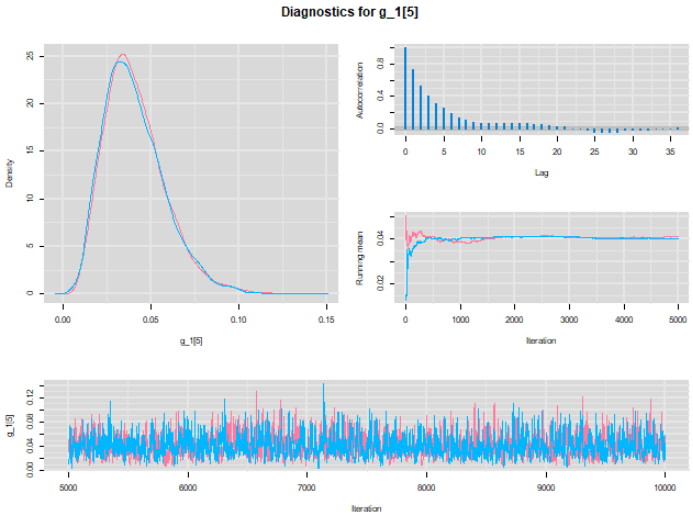
		sLong-DINA			sLong-LLM		
Núm.	Perfiles	*t=1*	*t=2*	*t=3*	*t=1*	*t=2*	*t=3*
1	000000	0.085	0.081	0.038	0.095	0.072	0.056
2	100000	0.072	0.060	0.041	0.141	0.106	0.107
3	010000	0.014	0.011	0.008	0.011	0.009	0.009
4	110000	0.041	0.039	0.035	0.077	0.072	0.075
5	001000	0.001	0.001	0.001	0	0	0
6	101000	0.003	0.003	0.003	0.003	0.003	0.003
7	011000	0.001	0.001	0.001	0	0	0
8	111000	0.005	0.005	0.007	0.009	0.010	0.010
9	000100	0.004	0.004	0.003	0.002	0.001	0.001
10	100100	0.014	0.013	0.012	0.015	0.016	0.016
11	010100	0.003	0.003	0.003	0.002	0.002	0.002
12	110100	0.024	0.024	0.031	0.047	0.050	0.054
13	001100	0	0	0	0	0	0
14	101100	0.002	0.002	0.003	0.003	0.003	0.003
15	011100	0.001	0.001	0.001	0	0	0
16	111100	0.013	0.010	0.016	0.039	0.031	0.033
17	000010	0	0	0	0	0	0
18	100010	0	0	0.001	0.001	0.001	0.001
19	010010	0	0	0	0	0	0
20	110010	0.001	0.001	0.002	0.002	0.003	0.003
21	001010	0	0	0	0	0	0
22	101010	0	0	0	0	0	0
23	011010	0	0	0	0	0	0
24	111010	0.002	0.001	0.002	0.003	0.003	0.003
25	000110	0	0	0	0	0	0
26	100110	0.001	0.001	0.001	0.001	0.001	0.001
27	010110	0	0	0	0	0	0
28	110110	0.006	0.004	0.006	0.018	0.015	0.015
29	001110	0	0	0	0	0	0
30	101110	0.001	0.001	0.001	0.003	0.003	0.002
31	011110	0	0	0	0	0	0
32	111110	0.032	0.024	0.027	0.126	0.114	0.114
33	000001	0.019	0.020	0.010	0.001	0	0
34	100001	0.028	0.026	0.019	0.003	0.002	0.002
35	010001	0.005	0.005	0.004	0	0	0
36	110001	0.026	0.023	0.025	0.004	0.004	0.004
37	001001	0	0	0	0	0	0
38	101001	0.002	0.002	0.002	0	0	0
39	011001	0	0	0	0	0	0
40	111001	0.006	0.005	0.008	0.002	0.001	0.001
41	000101	0.002	0.002	0.001	0	0	0
42	100101	0.010	0.009	0.010	0.001	0.001	0.001
43	010101	0.002	0.002	0.002	0	0	0
44	110101	0.028	0.024	0.036	0.009	0.008	0.008
45	001101	0	0	0	0	0	0

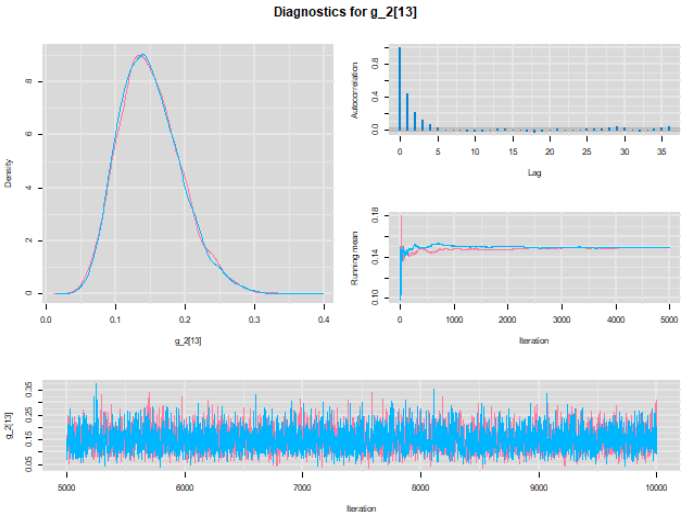
Tabla 7.3: Resultados de Proporciones combinadas de modelos no secuenciales (II)

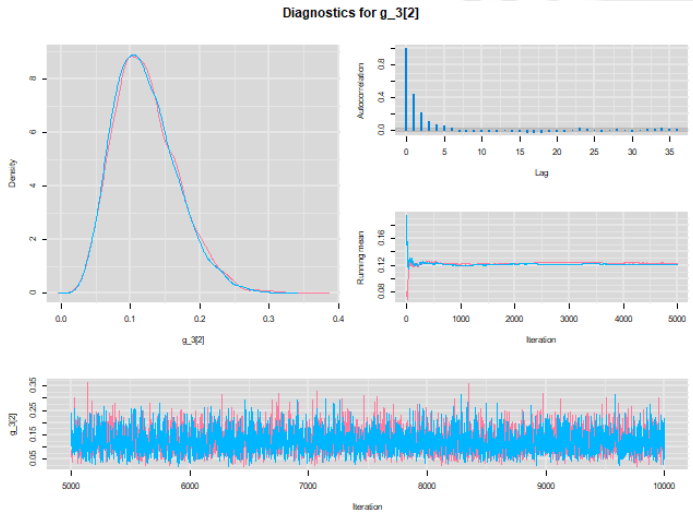
		sLong-DINA			sLong-LLM		
Núm.	Perfiles	*t=1*	*t=2*	*t=3*	*t=1*	*t=2*	*t=3*
46	101101	0.003	0.003	0.004	0.001	0.001	0.001
47	011101	0.001	0.001	0.001	0	0	0
48	111101	0.038	0.026	0.038	0.023	0.016	0.017
49	000011	0	0	0	0	0	0
50	100011	0	0	0	0	0	0
51	010011	0	0	0	0	0	0
52	110011	0.002	0.002	0.002	0.001	0.001	0.001
53	001011	0	0	0	0	0	0
54	101011	0	0	0	0	0	0
55	011011	0	0	0	0	0	0
56	111011	0.006	0.004	0.006	0.002	0.002	0.002
57	000111	0	0	0	0	0	0
58	100111	0.001	0.001	0.001	0	0	0
59	010111	0	0	0	0	0	0
60	110111	0.021	0.014	0.019	0.013	0.010	0.010
61	001111	0	0	0	0	0	0
62	101111	0.006	0.004	0.005	0.002	0.003	0.002
63	011111	0.002	0.002	0.002	0.001	0.001	0.001
64	111111	0.463	0.536	0.559	0.337	0.431	0.438

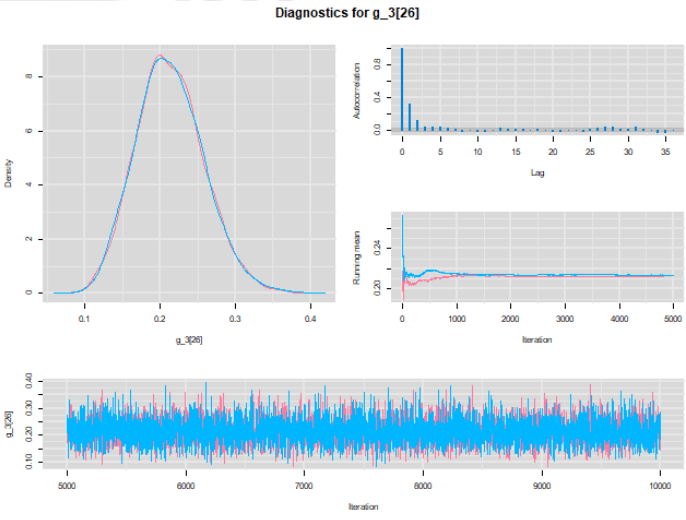
7.6. Anexo E. Plot de cadenas MCMC para estudio de Simulación

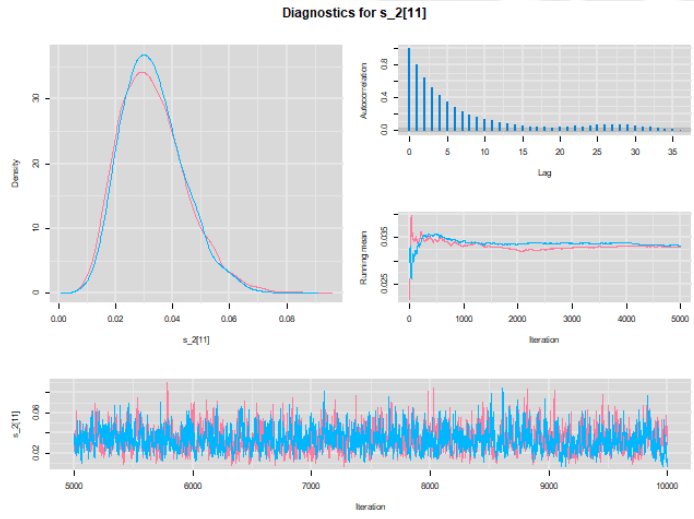
En este anexo, se agrega la verificación gráfica de las cadenas simuladas. Por brevedad, solo se muestran las que corresponden a la estructura lineal para ambos modelos Hi-Long-DINA y Hi-Long-LLM de los parámetros g_t, s_t y θ_t.

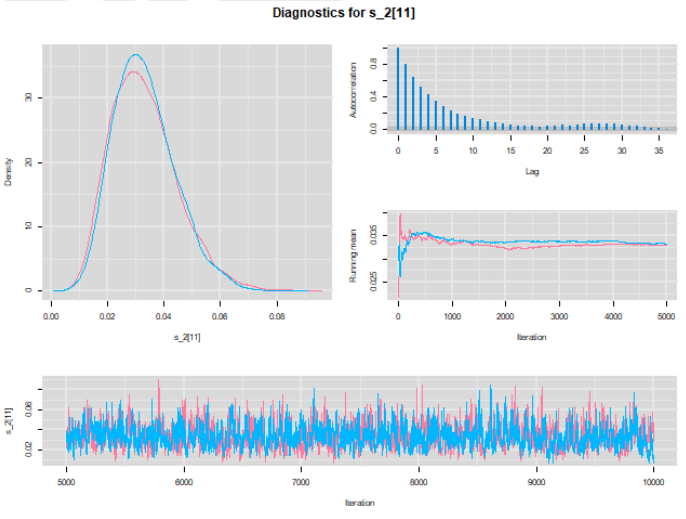

Diagnostics for g_1[5]
Density
g_1[5]
Autocorrelation
Lag
Running mean
Iteration
g_1[5]
Iteration

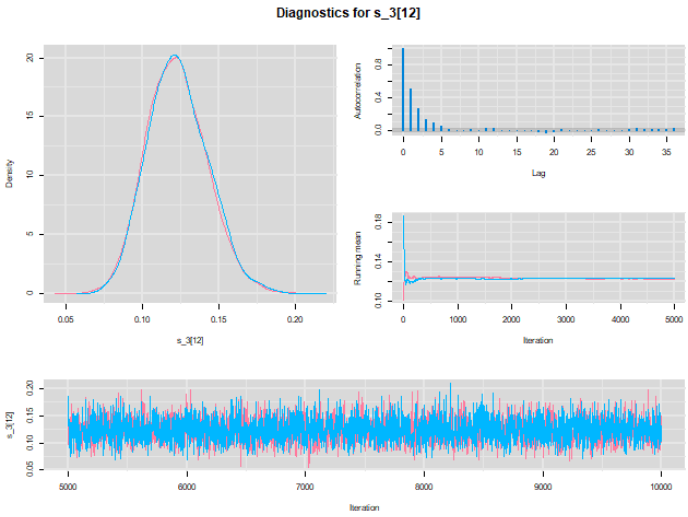

Diagnostics for g_2[13]
Density
g_2[13]
Autocorrelation
Lag
Running mean
Iteration
g_2[13]
Iteration

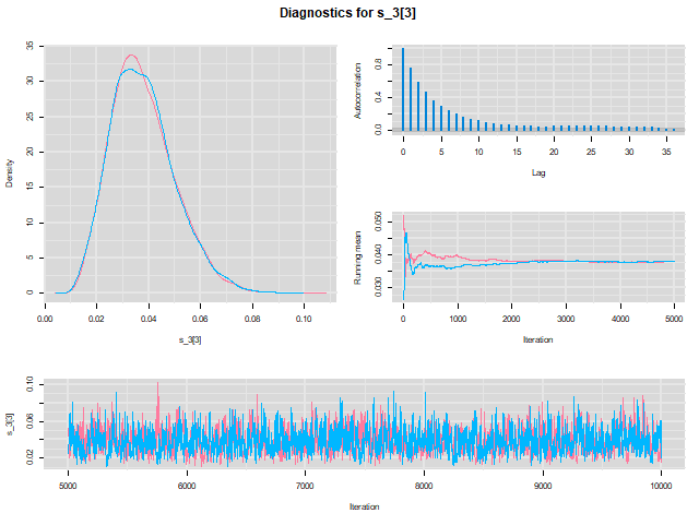

Diagnostics for g_3[2]
Density
g_3[2]
Autocorrelation
Lag
Running mean
Iteration
g_3[2]
Iteration

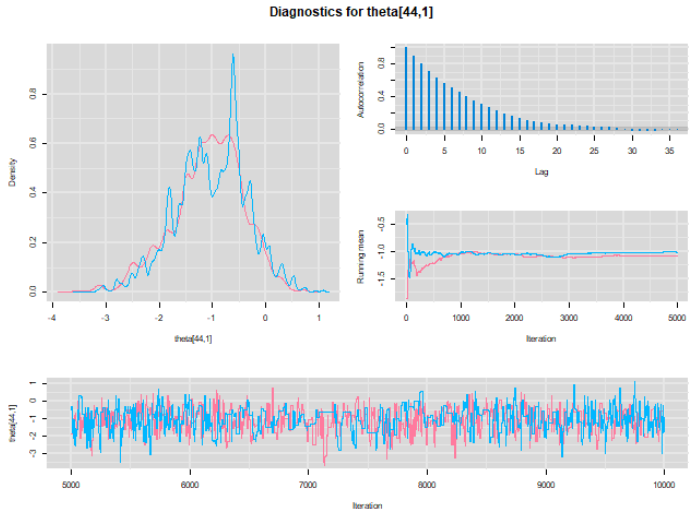

Diagnostics for g_3[26]
Density
g_3[26]
Autocorrelation
Lag
Running mean
Iteration
g_3[26]
Iteration

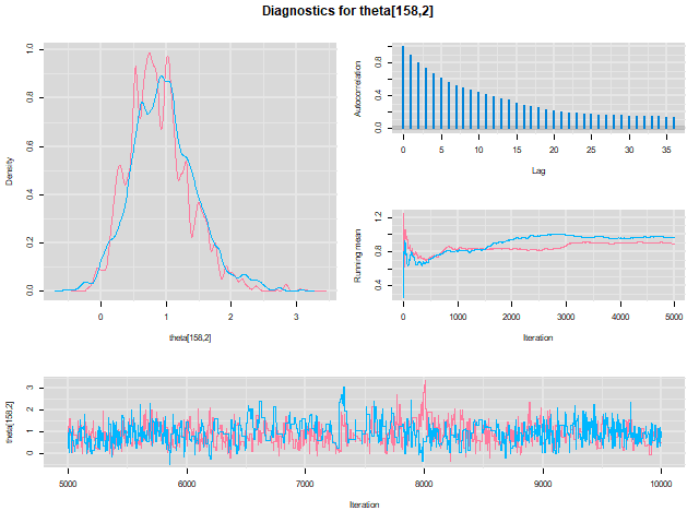

Diagnostics for s_2[11]
Density
s_2[11]
Autocorrelation
Lag
Running mean
Iteration
s_2[11]
Iteration

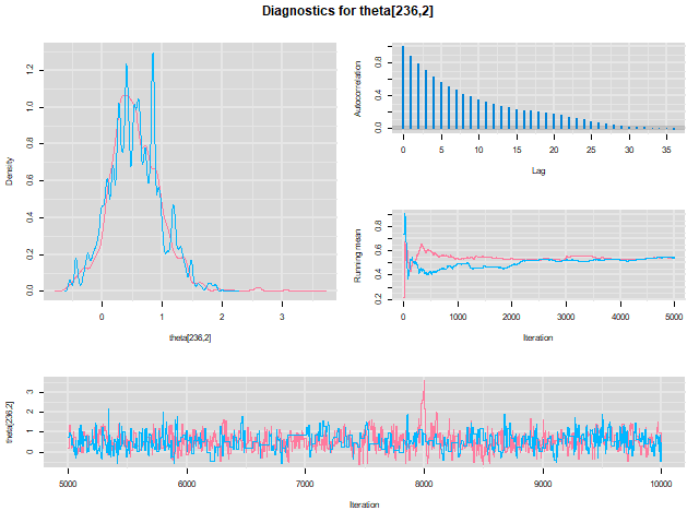

Diagnostics for s_2[11]
Density
s_2[11]
Autocorrelation
Lag
Running mean
Iteration
s_2[11]
Iteration

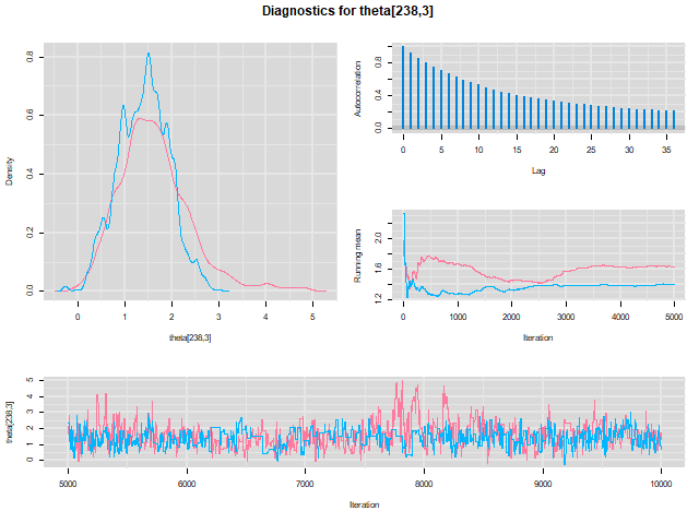

Diagnostics for s_3[12]
Density
s_3[12]
Autocorrelation
Lag
Running mean
Iteration
s_3[12]
Iteration

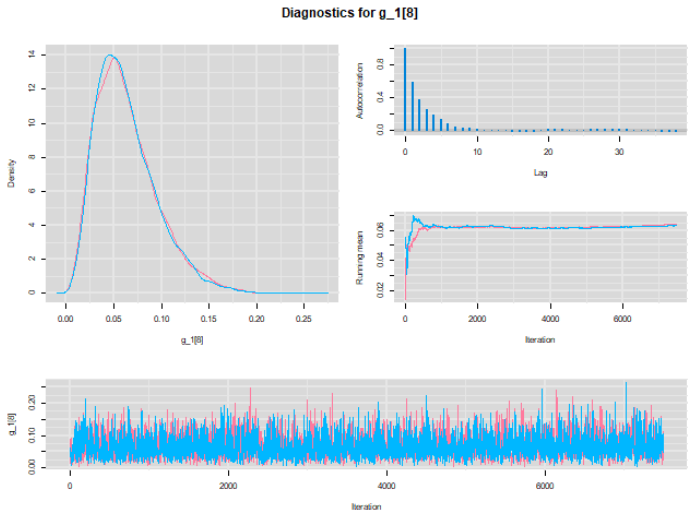

Diagnostics for s_3[3]
Density
s_3[3]
Autocorrelation
Lag
Running mean
Iteration
s_3[3]
Iteration

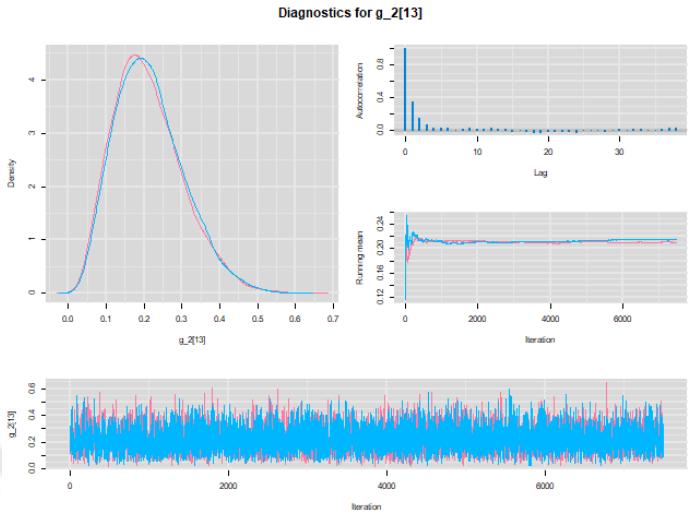

Diagnostics for theta[44,1]
Density
theta[44,1]
Autocorrelation
Lag
Running mean
Iteration
theta[44,1]
Iteration

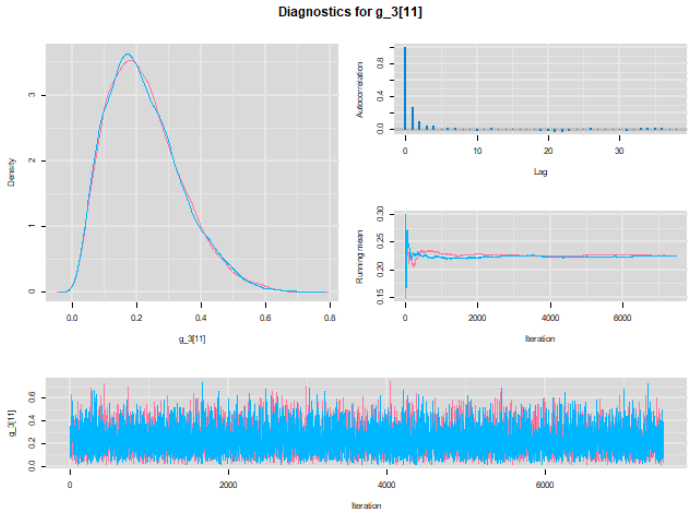

Diagnostics for theta[158,2]
Density
theta[158,2]
Autocorrelation
Lag
Running mean
Iteration
theta[158,2]
Iteration

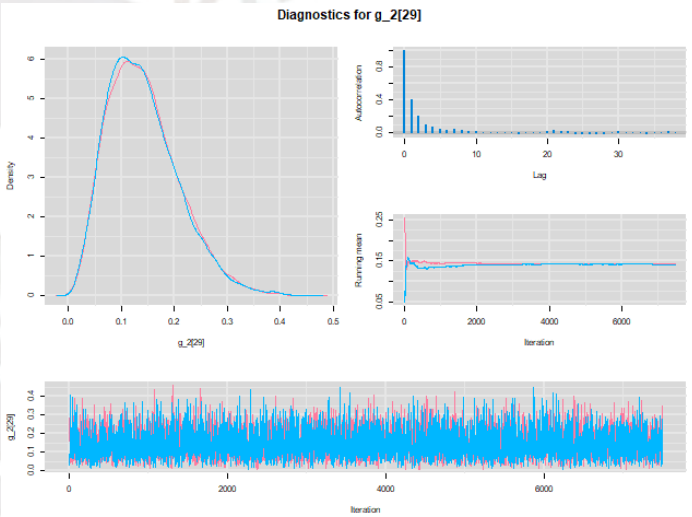

Diagnostics for theta[236,2]
Density
theta[236,2]
Autocorrelation
Lag
Running mean
Iteration
theta[236,2]
Iteration

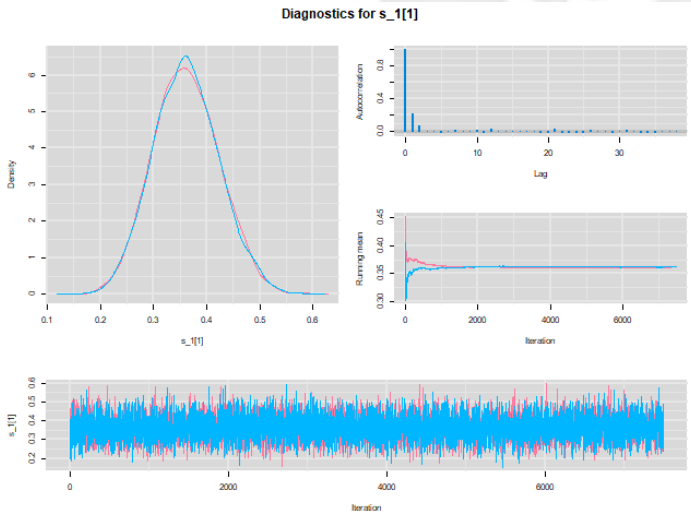

Diagnostics for theta[238,3]
Density
theta[238,3]
Autocorrelation
Lag
Running mean
Iteration
theta[238,3]
Iteration

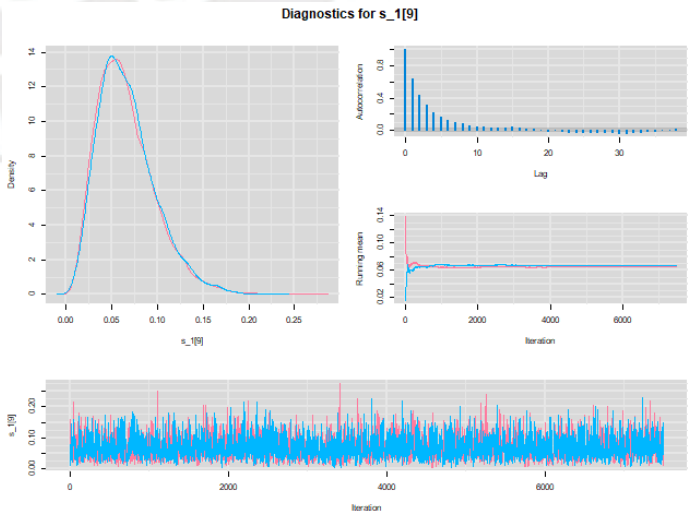

Diagnostics for g_1[8]
Density
g_1[8]
Autocorrelation
Lag
Running mean
Iteration
g_1[8]
Iteration

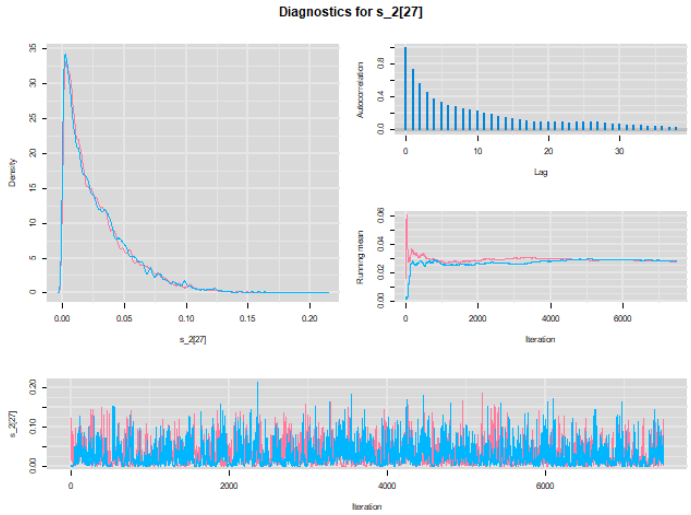

Diagnostics for g_2[13]
Density
g_2[13]
Autocorrelation
Lag
Running mean
Iteration
g_2[13]
Iteration

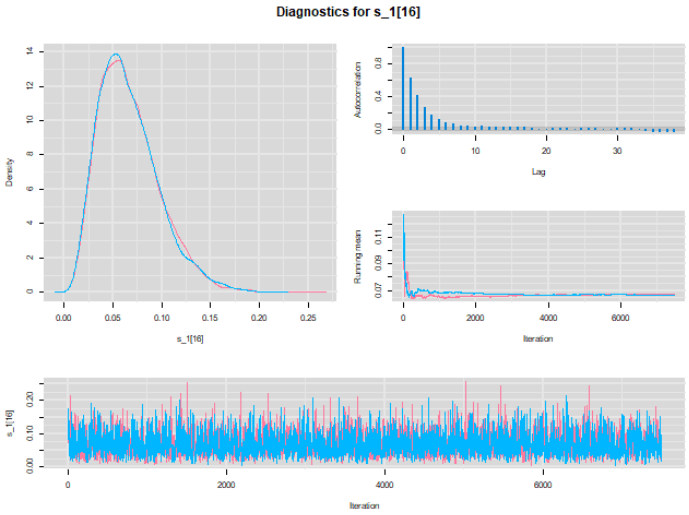

Diagnostics for g_3[11]
Density
g_3[11]
Autocorrelation
Lag
Running mean
Iteration
g_3[11]
Iteration

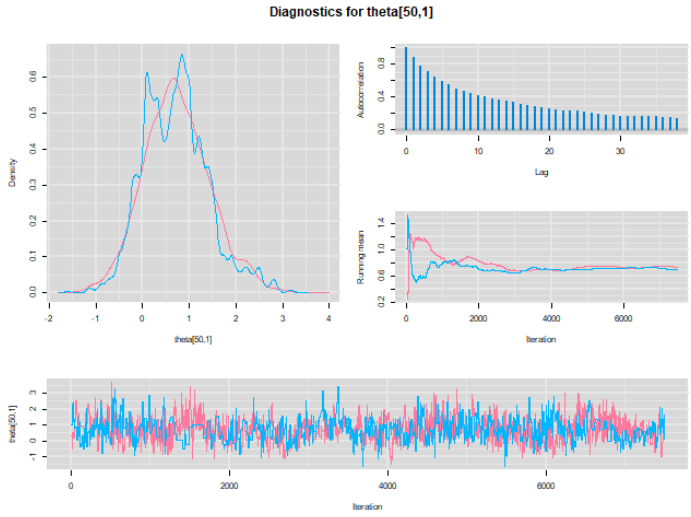

Diagnostics for g_2[29]
Density
g_2[29]
Autocorrelation
Lag
Running mean
Iteration
g_2[29]
Iteration

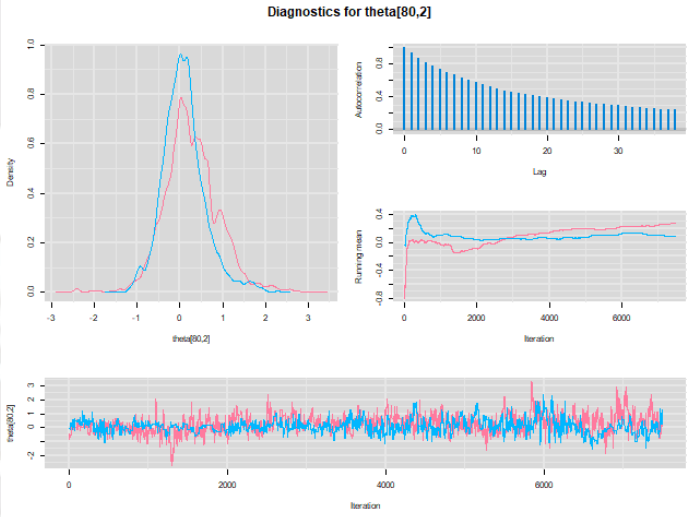

Diagnostics for s_1[1]
Density
s_1[1]
Autocorrelation
Lag
Running mean
Iteration
s_1[1]
Iteration

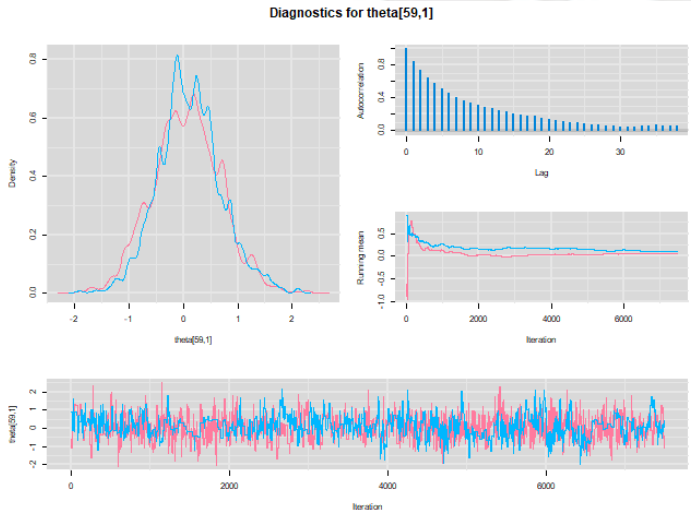

Diagnostics for s_1[9]
Density
s_1[9]
Autocorrelation
Lag
Running mean
Iteration
s_1[9]
Iteration

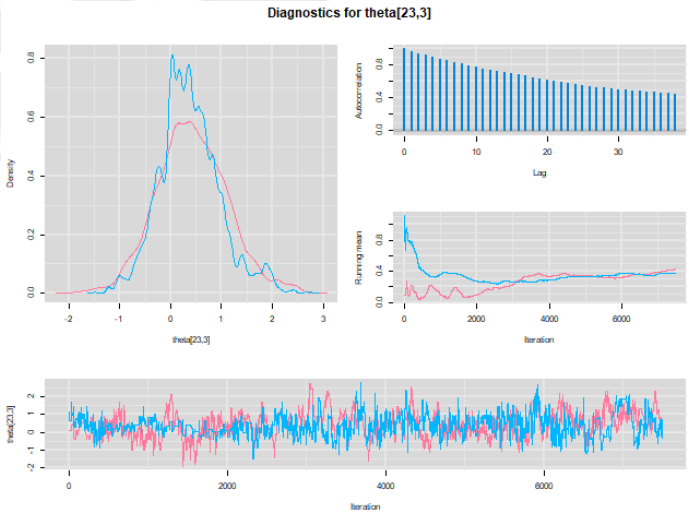

Diagnostics for s_2[27]
Density
s_2[27]
Autocorrelation
Lag
Running mean
Iteration
s_2[27]
Iteration


Diagnostics for s_1[16]
Density
s_1[16]
Autocorrelation
Lag
Running mean
Iteration
s_1[16]
Iteration


Diagnostics for theta[50,1]
Density
theta[50,1]
Autocorrelation
Lag
Running mean
Iteration
theta[50,1]
Iteration


Diagnostics for theta[80,2]
Density
theta[80,2]
Autocorrelation
Lag
Running mean
Iteration
theta[80,2]
Iteration


Diagnostics for theta[59,1]
Density
theta[59,1]
Autocorrelation
Lag
Running mean
Iteration
theta[59,1]
Iteration


Diagnostics for theta[23,3]
Density
theta[23,3]
Autocorrelation
Lag
Running mean
Iteration
theta[23,3]
Iteration

Bibliografía

Bartholomew, D., Knott, M. y Moustaki, I. (2011). *Latent Variable Models and Factor Analysis: A Unified Approach*.

Brooks, S. y Gelman, A. (1998). General methods for monitoring convergence of iterative simulations, *J. Comput. Graphi. Stat.* **7**: 434–455.

Cai, L. (2012). Latent variable modeling, *Shanghai archives of psychiatry* **24**: 118–20.

Davier, M. (2024). *Rasch Polytomous Models*, pp. 5820–5823.

Gelman, A., Carlin, J., Stern, H., Dunson, D., Vehtari, A. y Rubin, D. (2013). *Bayesian Data Analysis, Third Edition*, Chapman & Hall/CRC Texts in Statistical Science, Taylor & Francis.
URL: *https://books.google.com.pe/books?id=ZXL6AQAAQBAJ*

Gu, Y. y Xu, G. (2020). Identifiability of hierarchical latent attribute models.

Haertel, E. H. (1989). Using restricted latent class models to map the skill structure of achievement items, *Journal of Educational Measurement* **26**(4): 301–321.
URL: *https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1745-3984.1989.tb00336.x*

Henning, G. (1989). Meanings and implications of the principle of local independence, *Language Testing* **6**(1): 95–108.
URL: *https://doi.org/10.1177/026553228900600108*

Jabrayilov, R., Emons, W. y Sijtsma, K. (2016). Comparison of classical test theory and item response theory in individual change assessment, *Applied Psychological Measurement* **40**.

Junker, B. y Sijtsma, K. (2001). Cognitive assessment models with few assumptions, and connections with nonparametric item response theory, *Applied Psychological Measurement - APPL PSYCHOL MEAS* **25**: 258–272.

Kaya, Y. y Leite, W. L. (2017). Assessing change in latent skills across time with longitudinal cognitive diagnosis modeling: An evaluation of model performance, *Educational and psychological measurement* **77**(3): 369–388.

Kruschke, J. (2014). *Doing Bayesian data analysis: A tutorial with R, JAGS, and Stan, second edition.*

Lee, S. Y. (2017). Growth curve cognitive diagnosis models for longitudinal assessment. **URL:** *https://api.semanticscholar.org/CorpusID:126073538*

Leighton, J., Gierl, M. y Hunka, S. (2004). The attribute hierarchy method for cognitive assessment: A variation on tatsuoka's rule-space approach, *Journal of Educational Measurement* **41**.

Li, F., Cohen, A., Bottge, B. y Templin, J. (2015). A latent transition analysis model for assessing change in cognitive skills, *Educational and Psychological Measurement* **76**.

Likert, R. (1932). *A Technique for the Measurement of Attitudes*, number nos. 136-165 in *A Technique for the Measurement of Attitudes*, Columbia university. **URL:** *https://books.google.com.pe/books?id=9rotAAAAYAAJ*

Lin, Q., Xing, K. y Park, Y. S. (2020). Measuring skill growth and evaluating change: Unconditional and conditional approaches to latent growth cognitive diagnostic models, *Frontiers in Psychology* **11**: 2205.

Maris, E. (1995). Psychometric latent response models, *Psychometrika* **60**: 523–547.

Maris, E. (1999). Estimating multiple classification latent class models, *Psychometrika* **64**: 187–212.

Muthén, B. O. (1994). Multilevel covariance structure analysis, *Sociological methods & research* **22**(3): 376–398.

Muthén, B. (2002). Beyond sem: General latent variable modeling, *Behaviormetrika* **29**: 81–117.

Novick, M. R. (1980). Statistics as psychometrics, *Psychometrika* **45**(4): 411–424.

Pan, Q., Qin, L. y Kingston, N. (2020). Growth modeling in a diagnostic classification model (dcm) framework–a multivariate longitudinal diagnostic classification model, *Frontiers in Psychology* **11**: 1714.

Pan, Y. y Zhan, P. (2020). The impact of sample attrition on longitudinal learning diagnosis: A prolog, *Frontiers in psychology* **11**: 1051.

Plummer, M. (2003). Jags: A program for analysis of bayesian graphical models using gibbs sampling, *3rd International Workshop on Distributed Statistical Computing (DSC 2003); Vienna, Austria* **124**.

Rasch, G. (1960). *Probabilistic Models for Some Intelligence and Attainment Tests*, Studies in mathematical psychology, Danmarks Paedagogiske Institut.
URL: *https://books.google.com.pe/books?id=aB9qLgEACAAJ*

Spearman, C. (1904). "general intelligence,.ºbjectively determined and measured, *The American Journal of Psychology* **15**(2): 201–292.
URL: *http://www.jstor.org/stable/1412107*

Spiegelhalter, D., Best, N., Carlin, B. y Linde, A. (2002). Bayesian measures of model complexity and fit (with discussion), *Journal of the Royal Statistical Society, Series B* **64**: 1–34.

Tang, F. y Zhan, P. (2020). The development of an instrument for longitudinal learning diagnosis of rational number operations based on parallel tests, *Frontiers in Psychology* **11**.

Tang, F. y Zhan, P. (2021). Does diagnostic feedback promote learning? evidence from a longitudinal cognitive diagnostic assessment, *AERA Open* **7**: 1–15.

Templin, J. y Henson, R. (2006). Measurement of psychological disorders using cognitive diagnosis models. psychological methods, 11(3), 287-305, *Psychological methods* **11**: 287–305.

Torre, J. (2009). Dina model and parameter estimation: A didactic, *Journal of Educational and Behavioral Statistics - J EDUC BEHAV STAT* **34**: 115–130.

Torre, J. y Douglas, J. (2004). Higher-order latent trait models for cognitive diagnosis, *Psychometrika* **69**: 333–353.

Torre, J. y Minchen, N. (2014). Cognitively diagnostic assessments and the cognitive diagnosis model framework, *Psicología Educativa* **20**.

van Ravenzwaaij, D., Cassey, P. y Brown, S. (2016). A simple introduction to markov chain monte–carlo sampling, *Psychonomic Bulletin & Review* **25**.

Wang, Z. y Osterlind, S. (2013). *Classical Test Theory*, pp. 31–44.

Zhan, P. (2020). Longitudinal learning diagnosis: Minireview and future research directions, *Frontiers in Psychology* **11**.

Zhan, P. y He, K. (2021). A longitudinal diagnostic model with hierarchical learning trajectories, *Educational Measurement Issues and Practice* **40**: 18–30.

Zhan, P., Jiao, H., Liao, D. y Li, F. (2019). A longitudinal higher-order diagnostic classification model, *Journal of Educational and Behavioral Statistics* **2019**.

Zhan, P., ma, W., Jiao, H. y Ding, S. (2020). A sequential higher order latent structural model for hierarchical attributes in cognitive diagnostic assessments, *Applied Psychological Measurement* **44**: 65–83.