UNIVERSIDAD DE SALAMANCA

Departamento de Estadística

Tesis Doctoral

Análisis de Datos Acoplados: Modelo T3-PCA.

Autor: Elisa Frutos Bernal

Director: Mª Purificación Galindo Villardón

2014

Análisis de Datos Acoplados: Modelo T3-PCA.

Memoria que, para optar al Grado de Doctor
por el Departamento de Estadística de la
Universidad de Salamanca, presenta:

Elisa Frutos Bernal

Salamanca, 2014

Mª PURIFICACIÓN GALINDO VILLARDÓN

Profesora Titular del Departamento de Estadística de la Universidad de Salamanca

---

CERTIFICA:

Que Elisa Frutos Bernal, Licenciada en Matemáticas, ha realizado en el Departamento de Estadística de la Universidad de Salamanca, bajo su dirección, el trabajo que para optar al Grado de Doctor, presenta con el título: **Análisis de Datos Acoplados: Modelo T3-PCA** y para que conste, firma el presente certificado en Salamanca, a 2 de Diciembre de 2014.

Dra. Mª Purificación Galindo Villardón

“El futuro tiene muchos nombres.
Para los débiles es lo inalcanzable.
Para los temerosos, lo desconocido.
Para los valientes es la oportunidad.“
(Victor Hugo)

AGRADECIMIENTOS

En primer lugar a la **Dra. Mª Purificación Galindo Villardón**, directora de este trabajo, por su dedicación continua, por sus consejos (no sólo estadísticos), por la confianza que ha demostrado siempre en mí y por haber sabido tirar de mi en todos esos momentos de bajón en los que me falta confianza en mi misma. Formas parte de mi vida no sólo como mi profesora, sino también como mi amiga. Gracias Puri...

A todos los profesores del **Departamento de Estadística** por sus enseñanzas en el doctorado y por hacerme sentir siempre como uno más.

A mi compi de fatigas **Ana Nieto Librero** por su apoyo, ayuda y amistad en todo este tiempo.

Al **Dr. Tom Wilderjans** de la Universidad de Leuven (Bélgica) por su ayuda desinteresada y su total disponibilidad para responder a cualquier duda que me haya podido surgir.

A **mis padres**, que haciendo grandes esfuerzos me han dado todo lo necesario para que hoy pueda estar aquí.

A **Carlos** por apoyarme en todos mis proyectos y estar ahí en los momentos en los que todo parecía negro. Por eso y por mucho más...

A **mis hermanas** por cuidar siempre de su hermana pequeña y ayudarme en todo lo que está en sus manos. Gracias por vuestro apoyo.

A **mis compañeros del Departamento de Matemáticas del IES Torres Villarroel**, por sus sabios consejos en los ratillos del café.

A **Marian** por su continua ayuda con el inglés, mi pequeña cruz...

Agradezco a todas las personas que de una u otra forma contribuyeron a la realización de esta tesis doctoral.

Índice general

NOTACIÓN

Con el fin de hacer más clara la exposición se ha introducido una notación que se va a seguir a lo largo de toda la tesis.

$\mathbf{Y}_{(r)}$ - matriz de datos de orden $(n \times m)$ y rango r

$\mathbf{Y}_{(q)}$ - es la aproximación de la matriz $\mathbf{Y}$ de rango q $(q < r)$

$\mathbf{U}_{(q)}$ - es una matriz de orden $(n \times q)$, cuyos vectores columna son los vectores singulares por la izquierda de $\mathbf{YY}^{'}$, los cuales son ortonormales, por lo que $\mathbf{UU}^{'} = \mathbf{I}$

$\mathbf{V}_{(q)}$ - es una matriz de orden $(m \times q)$, cuyos vectores columna son los vectores singulares por la derecha de $\mathbf{Y}^{'}\mathbf{Y}$, ortonormales, luego $\mathbf{V}^{'}\mathbf{V} = \mathbf{I}$

$\boldsymbol{\lambda}_{(q)}$ - es una matriz diagonal que contiene los q mayores valores singulares α_k asociados a los λ_k valores propios de la matriz $\mathbf{Y}^{'}\mathbf{Y}$, $(\alpha_1 \geq \alpha_2 \geq \cdots \geq \alpha_q)$

$\boldsymbol{G}_{(q)}$ - es la aproximación de rango q de la matriz $\boldsymbol{G}$ $(n \times q)$ de marcadores fila

$\boldsymbol{H}_{(q)}$ - es la aproximación de rango q de la matriz $\boldsymbol{H}$ $(m \times q)$ de marcadores columna

$\mathbf{X}$ - matriz de datos de dos dimensiones $I \times J$

x_{ij} - valor de la medida para el individuo i en la variable j

$\underline{\mathbf{X}}$ - matriz de datos de tres vías $I \times J \times K$

x_{ijk} - valor de la medida para el individuo i la variable j y la condición k

i - individuos; $i = 1, ..., I$

j - variables; $j = 1, ..., J$

k - condiciones; $k = 1, ..., K$

I - número total de elementos del primer modo (individuos)

J - número total de elementos del segundo modo (variables)

K - número total de elementos del tercer modo (condiciones)

p - número de componentes seleccionadas para describir el primer modo del array de datos (individuos); $p = 1, ..., P$

q - número de componentes seleccionadas para describir el segundo modo del array de datos (variables); $q = 1, ..., Q$

r - número de componentes seleccionadas para describir el tercer modo del array de datos (condiciones); $r = 1, ..., R$

P - número total de componentes del primer modo (individuos)

Q - número total de componentes del segundo modo (variables)

R - número total de componentes del tercer modo (condiciones)

S - número total de componentes $(P + Q + R)$

$\mathbf{A}$ - matriz de componentes $I \times P$ con los coeficientes de los individuos (primer modo)

$\mathbf{B}$ - matriz de componentes $J \times Q$ con los coeficientes de las variables (segundo modo)

$\mathbf{C}$ - matriz de componentes $K \times R$ con los coeficientes de las condiciones (tercer modo)

a_{ip} - elementos de la matriz de componentes $\mathbf{A}$ $(I \times P)$

b_{jq} - elementos de la matriz de componentes $\mathbf{B}$ $(J \times Q)$

c_{kr} - elementos de la matriz de componentes $\mathbf{C}$ $(K \times R)$

$\underline{\mathbf{G}}$ - core matrix $(P \times Q \times R)$

g_{pqr} - elementos de la core matrix

$\otimes$ - producto de Kronecker

$\bigodot$ - producto de Khatri-Rao

$\underline{\mathbf{D}}^1$ y $\mathbf{D}^2$ - matrices de datos de tamaños $(I \times J \times K)$ y $(I \times L)$ respectivamente que tienen un modo en común (en este caso el primero)

$\underline{\mathbf{M}}^1$ y $\mathbf{M}^2$ - matrices del modelo de tamaños $(I \times J \times K)$ y $(I \times L)$ respectivamente

$\mathbf{A}$, $\mathbf{B}^1$ y $\mathbf{C}$ - matrices de componentes del modelo $PARAFAC/Tucker3$ para $\underline{\mathbf{M}}^1$

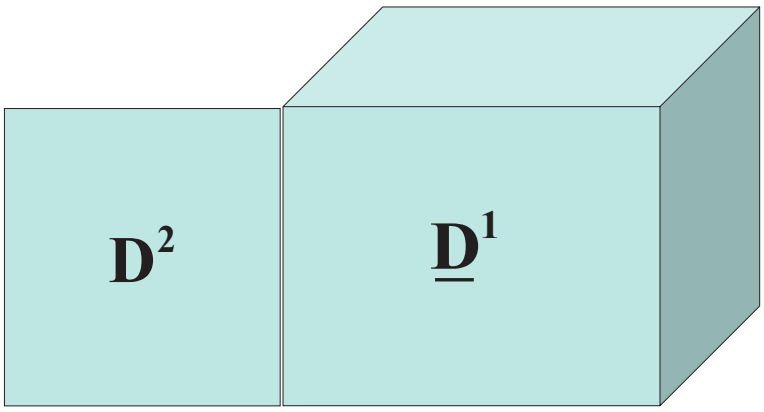
$\mathbf{A}$, $\mathbf{B}^2$ - matrices de la descomposición PCA para $\mathbf{M}^2$.

RESUMEN

INTRODUCCIÓN

En esta sociedad de la información en la que nos encontramos es bastante habitual encontrarnos con conjuntos de datos que incluyen distintos tipos de información sobre el mismo sistema, datos que se conocen con el nombre de *Coupled*, *Linked* o *Multiset Data*. Ejemplos de este tipo de datos podemos encontrarlos en psicometría, neuropsicología, bioinformática y quimiometría.

Dichos datos se pueden definir mediante un conjunto de bloques de N vías, de manera que cada uno de los bloques tiene al menos un modo en común con al menos otro bloque de datos Wilderjans et al. (2009b).



Representación gráfica de un conjunto de Datos Acoplados

Para descubrir la estructura subyacente al conjunto de Datos Acoplados, se puede representar cada bloque de datos de N vías mediante un modelo, de manera que los parámetros de los modelos sean iguales para cada modo común.

Para representar el conjunto de Datos Acoplados se pueden utilizar dos estrategias: (i) La primera estrategia se conoce con el nombre de estrategia segmentada, y consta de dos pasos: En primer lugar, se ajusta un modelo al bloque de datos de mayor interes; y en segundo lugar se utiliza la cuantificación del modo común obtenida en el paso anterior para el análisis del otro bloque de datos. Por lo tanto, existen diferentes funciones de

pérdida, una para cada modelo, y se optimizan cada una de ellas de forma separada (ii) La segunda estrategia se conoce con el nombre de integrada dado que existe una única función de pérdida global a optimizar. Esta función puede ser la suma ponderada de las funciones de pérdida individuales, una para cada submodelo. Los pesos indican el grado en el que cada bloque de datos influye en las estimación de los parámetros del modo o modos comunes.

Es de esperar que la estrategia integrada sea mejor que la segmentada sobre todo cuando los bloques de información además de tener información común tienen una cantidad considerable de información distintiva. En este caso, la estrategia segmentada se verá influida negativamente por la información distintiva y revelará peor la información común, mientras que la estrategia integrada recuperará mejor la información común.

Dos de los modelos más utilizados para el Análisis Integrado de Datos Acoplados son los modelos $PARAFAC$-PCA (Wilderjans et al., 2009a) y el modelo $CHIC$ (Wilderjans et al., 2008).

Para Datos Acoplados de tipo binario, bloque de tres vías y bloque de dos vías que comparten un modo, tenemos el modelo $CHIC$. En este modelo el bloque de tres vías se ajusta según un modelo $INDCLAS$ (Leenen et al. (1999)) mientras que el bloque de dos vías se ajusta según un modelo $HICLAS$ (De Boeck and Rosenberg (1988); Van Mechelen et al. (1995); Leenen et al. (1999, 2001)). Ambos submodelos se relacionan haciendo que la matriz de componentes del modo común sea igual en ambos submodelos.

Se puede utilizar una estrategia integrada para analizar un bloque de datos de tres vías y un bloque de datos de dos vías, ambos de tipo continuo, que tienen un modo en comun. Este modelo, conocido con el nombre de $PARAFAC$-PCA, representa el bloque de datos de tres vías mediante un modelo $PARAFAC$ (Carroll and Chang, 1970; Harshman, 1970) y el bloque de dos vías mediante un modelo PCA. Ambos submodelos se relacionan mediante la imposición de la restricción de que la matriz de componentes del modo común ha de ser igual en ambos submodelos. El modelo $PARAFAC$ es bastante restrictivo ya que no permite la interacción entre todas las componentes del modelo, de ahi que en esta tesis se presente el modelo $T3$-PCA.

El modelo $T3$-PCA es útil para el análisis de un bloque de datos de tres vías y un bloque de datos de dos vías, ambos de tipo continuo, que tienen un modo común. Más concretamente, queremos construir un modelo que conste de dos submodelos, un modelo $Tucker3$ Tucker (1966); Kroonenberg and de Leeuw (1980) para el bloque de datos de tres vías y un modelo PCA para el bloque de dos vías. Los submodelos se relacionan imponiendo la restricción de que la matriz de componentes del modo común ha de ser igual en ambos submodelos.

Tanto el nuevo modelo propuesto para el Análisis de Datos Acoplados $T3$-PCA como los mencionados anteriormente utilizan distintos modelos para el Análisis de la Interacción en bloques de datos de dos y tres vías, de ahí que la tesis comience con una amplia revisión bibliográfica sobre el Análisis de la Interacción en Tablas de dos y tres vías de tipo continuo y binario.

OBJETIVOS

Los objetivos de esta investigación son:

(1) Hacer una revisión de los métodos para el Análisis de la Interacción en tablas de dos vías y presentar el paquete `GGEBiplotGUI` desarrollado en `R` que posibilita el Análisis de la Interacción en Tablas de dos vías.

(2) Hacer una revisión de los métodos para el análisis de la interacción en tablas de tres vías con datos de tipo real, centrándome fundamentalmente en los modelos $CANDECOMP/PARAFAC$ y $Tucker3$.

(3) Hacer una revisión de los métodos para el análisis de tablas de tres vías con datos de tipo binario, especialmente de los modelos $INDCLAS$ y $Tucker3$-$HICLAS$.

(4) Estudiar la aplicación de los métodos anteriores en el análisis de tablas de dos vías con datos tipo rating, revisión del modelo $HICLAS$-R.

(5) Estudio de estrategias existentes para el Análisis de Datos Acoplados. Revisión de los modelos $PARAFAC$-PCA y $CHIC$.

(6) Proponer un nuevo modelo para el Análisis de Datos Acoplados de tipo continuo. Además de presentarse el modelo se propondrá un algoritmo para el cálculo de los parámetros del modelo, un método para el cálculo del rango del mismo y rotaciones que mejoren la interpretabilidad de los resultados.

(7) Realizar un estudio de simulación que permita evaluar el rendimiento del algoritmo propuesto desde el punto de vista de la optimización del mismo y la capacidad para la recuperación de datos. Además se comparará el comportamiento de este algoritmo con el de otras estrategias (segmentadas y separadas).

(8) Facilitar el software necesario para realizar el Análisis Segmentado e Integrado de Datos Acoplados de tipo continuo.

ESTRUCTURA DE LOS CONTENIDOS

La parte principal de la tesis se divide en siete capítulos:

El **CAPÍTULO 1** comienza con una revisión de los métodos biplot: biplot clásicos (Gabriel, 1971), HJ biplot (Galindo, 1986; Galindo and Cuadras, 1986) y GGE biplot (Yan et al., 2000). Es al introducir el GGE biplot cuando se hace una revisión de los métodos existentes para el Análisis de la Interacción en tablas de dos vías. Además se presenta una paquete desarrollado en el lenguaje `R` que implementa las principales funciones de los biplot clásicos y del GGE biplot. La importancia de dicho paquete radica en que es el primer paquete en software libre que permite construir GGE biplots, además es interactivo por lo que para su uso no se requiere ningún conocimiento de `R`. El capítulo finaliza con una aplicación a datos utilizando el paquete `GGEbiplotGUI` anteriormente mencionado.

En el capítulo **ARTICULOS** de la tesis se incluye el artículo publicado en 2013 en la Revista **STOCHASTIC ENVIRONMENTAL RESEARCH AND RISK ASSESSMENT** (con índice de impacto 2.673, situada en el primer cuartil de la categoria

STATISTICS and PROBABILITY. Ranking: 6/119), y que forma parte del trabajo realizado para la elaboración de esta tesis: **"An interactive biplot implementation in R for modeling genotype-by-environment interaction"**.

En el **CAPÍTULO 2** se realiza una exhaustiva revisión de los Modelos para Tablas de Tres Vías con datos continuos. Los modelos que se abordan con mayor detalle son los Modelos $CANDECOMP/PARAFAC$ y los Modelos $Tucker$. Para cada de uno de ellos se recoge su algoritmo correspondiente, los métodos para la selección del número de componentes que se han de retener y aplicaciones de dichos modelos.

En el **CAPÍTULO 3** se realiza una exhaustiva revisión de los Modelos para Tablas de Tres Vías con datos binarios. Para ello se introducen los Modelos de Clases Jerárquicas para Tablas de Dos Vías Binarias y a partir de estos se presentan los Modelos $INDCLAS$ y $Tucker3$-$HICLAS$ para el análisis de Tres Vías. Además se hace una pequeña aplicación a datos utilizando el modelo $Tucker3$-$HICLAS$.

En el **CAPÍTULO 4** se presenta el modelo $HICLAS$-R para el Análisis de Tablas de Dos Vías con datos tipo rating. Este modelo supone una extensión de los Modelos de Clases Jerárquicas introducidos en el Capítulo anterior y es una alternativa a la dicotomización que habitualmente se utiliza para el tratamiento de este tipo de datos. Se presenta acompañado de una pequeña aplicación a datos que ilustra el mismo.

El **CAPÍTULO 5** se dedica al Análisis de Datos Acoplados presentándose la Familia de Modelos de Componentes Multivía-Multibloque. Se realiza un estudio de los modelos $PARAFAC$-PCA y $CHIC$ para el análisis de una tabla de tres vías y una de dos vías que tienen un modo en común.

En el **CAPÍTULO 6** se presenta el nuevo modelo, $T3$-PCA, que mediante una estrategia integrada permite el análisis de una matriz de tres vías y una matriz de dos vías, ambas con datos de tipo continuo, que tienen un modo en común. En este capítulo además de proponerse dicho modelo se presenta el algoritmo para el cálculo de los parámetros del modelo y un método para la selección del rango del mismo. Además se lleva a cabo un estudio de simulación para comparar el rendimiento de dicha estrategia con otras estragias (segmentadas y separadas). Finalmente se presenta una aplicación

del modelo $T3$-PCA a datos reales. También se proponen algunas líneas de investigación futuras de dicho modelo.

En el capítulo **ARTÍCULOS** de la tesis se incluye el artículo sometido a la Revista **BEHAVIOR RESEARCH METHODS** (con índice de impacto 2.458), y que forma parte del trabajo realizado para la elaboración de esta tesis: **"Data fusion by T3-PCA: An integrated global model for the simultaneous analysis of coupled real-valued data"**.

En el **CAPÍTULO 7** se recoge el software que se ha desarrollado en `R` para poder llevar a cabo el estudio de simulación que se presenta en el capítulo anterior, además del análisis de datos según el algorítmo $T3$-PCA.

ASPECTOS MÁS RELEVANTES DEL TRABAJO

La forma típica de presentar los datos para el análisis estadístico es formando una tabla de dos vías; es decir, en una matriz $\mathbf{X}$ de dimensión $(I \times J)$ donde I representa las unidades de muestreo/objetos/filas y J el número de variables/columnas. Existen muchos modelos en la literatura para capturar la información recogida en la matriz de dos vías, en concreto para el análisis de la interacción en tablas de dos vías.

El biplot se ha convertido en una herramienta de visualización de datos muy popular en muchas áreas de investigación, tales como la psicología, la medicina, la sociología, la ecología y la agricultura. La primera aplicación del biplot en datos de agricultura la hicieron Bradu and Gabriel (1978) y fue para ilustrar el uso del biplot como herramienta de diagnóstico en la selección de modelos.

En los últimos años ha aparecido mucha literatura acerca del uso del Análisis Biplot en el estudio de la interacción. Este análisis biplot se realiza sobre uno de los diferentes modelos lineales-bilineales existentes. Los modelos más comunes en el análisis biplot de la interacción han sido el modelo 'The additive main effects and multiplicative interaction' (AMMI) y el modelo 'Genotype main effects and genotype x environment interaction effects' (GGE) (Gauch, 1988).

En los modelos AMMI se introducen tantos términos multiplicativos como sean necesarios para explicar la interacción. Estos modelos se basan en la Descomposición en Valores y Singulares (SVD) de la matriz de residuales del modelo. El modelo GGE aplica la descomposición SVD de los datos a los que se les sustraido previamente el efecto columna. De manera que el GGE biplot visualiza el efecto principal fila y la interacción.

En el análisis de datos de tres vías la información se presenta en arreglos donde cada dato se indexa con tres índices: uno que indentifica al individuo i $(i = 1, ..., I)$, otro a la variable j $(j = 1, ..., J)$ y un tercero que corresponde a la condición k $(k = 1, ..., K)$; representándose en un arreglo tridimensional: individuos, variables y condiciones.

Existen distintos modelos para el Análisis de la Interacción en tablas de tres vías con datos continuos. Los más utilizados son el modelo $CANDECOMP/PARAFAC$ Harshman (1970); Carroll and Chang (1970) y el modelo $Tucker3$ (Tucker, 1966).

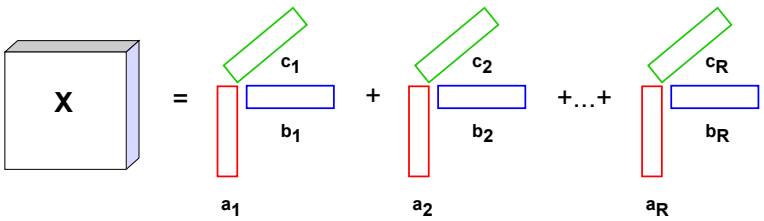
La descomposición $CANDECOMP/PARAFAC$ de una matriz $\underline{\mathbf{X}} \in \Re^{I \times J \times K}$ es de la forma:

$$\underline{\mathbf{X}} \simeq \sum_{r=1}^{R} a_r \circ b_r \circ c_r$$

con $a_r \in \Re^I$, $b_r \in \Re^J$ y $c_r \in \Re^K$ para $r = 1, ..., R$. Equivalentemente:

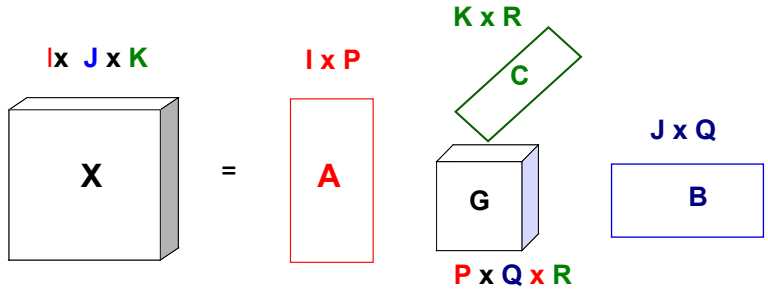
$$x_{ijk} \simeq \sum_{r=1}^{R} a_{ir} b_{jr} c_{kr}$$

con $i = 1, ..., I$, $j = 1, ..., J$, $k = 1, ..., K$ y R el rango del modelo.



Descomposición CP de una matriz de tres vías.

.



Descomposición *Tucker* de una matriz de tres vías.

La descomposición *Tucker* fue introducida por Tucker (1966) y es una forma de *PCA* de orden superior. Dada una matriz de tres vías donde $\underline{\mathbf{X}} \in \Re^{I \times J \times K}$ la descomposición *Tucker*3 de rango (P, Q, R) sería:

$$\underline{\mathbf{X}} \simeq \sum_{p=1}^{P} \sum_{q=1}^{Q} \sum_{r=1}^{R} g_{pqr} a_p \circ b_q \circ c_r$$

Donde $\mathbf{A} \in \Re^{I \times P}$, $\mathbf{B} \in \Re^{J \times Q}$ y $\mathbf{C} \in \Re^{K \times R}$ pueden entenderse como las componentes principales de cada modo. La matriz $\underline{\mathbf{G}} \in \Re^{P \times Q \times R}$ se llama core matrix y **muestra el nivel de interacción entre las diferentes componentes**.

P, Q y R son el número de componentes (es decir, columnas) de las matrices **A**, **B** y **C** respectivamente.

Por tanto la diferencia fundamental entre ambos modelos radica en la existencia de la core matrix, que recoge las interacciones entre las distintas componentes.

Para la representación de datos binarios en N-vías podemos considerar la familia de **modelos de clases jerárquicas** (*HICLAS*) (De Boeck and Rosenberg (1988); Van Mechelen et al. (1995); Leenen et al. (1999, 2001)). Estos modelos **no sólo incluyen una regla de descomposición sino que también representan relaciones de equivalencia y jerarquía** entre los elementos de cada modo.

En estos modelos nos vamos a referir a los elementos del primer modo como **objetos**,

a los del segundo modo como **atributos** y a los del tercer modo como **recursos**.

Dentro de la familia de modelos de clases jerárquicas tenemos el modelo $INDCLAS$ (modelo jerárquico de clases de diferencias individuales) (Leenen et al. (1999))y el modelo $Tucker3$-$HICLAS$ (Ceulemans et al. (2003)).

El conjunto de relaciones teóricas de equivalencia y jerarquía son idénticas para los modelos $INDCLAS$ y $Tucker3$-$HICLAS$, sólo difieren en la relación de asociación.

El modelo $INDCLAS$ (Leenen et al., 1999) aproxima un array de datos binario $\underline{\mathbf{D}}$ de tamaño I (objetos) x J (atributos) x K (recursos) mediante un array binario $\underline{\mathbf{M}}$ ($I \times J \times K$) que puede ser descompuesto en una matriz binaria $\mathbf{A}$ ($I \times R$), una matriz binaria $\mathbf{B}$ ($J \times R$) y una matriz binaria $\mathbf{C}$ ($K \times R$), donde R denota el rango del modelo. Las R columnas de $\mathbf{A}$, $\mathbf{B}$ y $\mathbf{C}$ definen R clusters, llamados paquetes ("bundles") de objetos, atributos y recursos, respectivamente.

En el modelo $INDCLAS$ la relación de asociación (que es la relación ternaria entre objetos, atributos y recursos y se define como una entrada 1 en el array $\underline{\mathbf{M}}$) se representa mediante la siguiente regla:

$$m_{ijk} = \bigoplus_{r=1}^{R} a_{ir} b_{jr} c_{kr} (\forall i = 1...I, \forall j = 1...J, \forall k = 1...K)$$

donde $\bigoplus$ se corresponde con la suma booleana. En resumen, un elemento m_{ijk} es igual a uno si y solo si existe un paquete al que pertenece el objeto i, el atributo j y el recurso k.

El modelo $Tucker3$-$HICLAS$ (Ceulemans et al. (2003)) implica una descomposición de un array binario $\underline{\mathbf{M}}$ de tamaño $I \times J \times K$ en una matriz binaria $\mathbf{A}$ de objetos de tamaño $I \times R$, una matriz binaria $\mathbf{B}$ de atributos de tamaño $J \times S$, una matriz binaria $\mathbf{C}$ de recursos de tamaño $K \times T$ y una matriz binaria $\mathbf{G}$ de tamaño $R \times S \times T$ siendo (R, S, T) el rango del modelo. $\mathbf{G}$ define las relaciones de asociación entre los objetos, atributos y recursos; a esta matriz se le llama core matrix.

En el modelo $Tucker3$-$HICLAS$ la relación de asocición se define como sigue:

$$m_{ijk} = \bigoplus_{r=1}^{R} \bigoplus_{s=1}^{S} \bigoplus_{t=1}^{T} a_{ir} b_{js} c_{kt} g_{rst} (\forall i = 1...I, \forall j = 1...J, \forall k = 1...K)$$

Es decir m_{ijk} es igual a 1 si el objeto i, el atributo j y el recurso k pertenecen a un paquete objeto, paquete atributo y paquete recurso y estos estan asociados en la core matrix $\underline{\mathbf{G}}$. Si se reemplaza el $\bigoplus$ en la ecuación anterior por una suma se obtiene el modelo $Tucker3$ (Tucker (1966)), por lo que el modelo $Tucker3$-$HICLAS$ es el modelo de clases jerárquico homólogo al modelo $Tucker3$. El modelo $INDCLAS$ es el modelo de clases jerárquico homólogo al modelo $PARAFAC$.

Van Mechelen et al. (2007) presenta una extensión de los modelos de clases jerárquicas denominado $HICLAS$-R para datos de clasificación. Este modelo incluye una generalización de las relaciones de equivalencia, jerarquía y asociación, y además incluye una representación gráfica completa.

Hasta aquí lo relativo al análisis de la interacción en tablas de dos y tres vías, sin embargo en los diferentes campos de investigación es muy común encontrarnos con conjuntos de bloques de datos que tienen uno o más modos en común, estos datos se conocen con el nombre de datos acoplados.

Son varios los modelos existentes para el análisis de este tipo de datos, nosotros en esta tesis presentamos los modelos $CHIC$ Wilderjans et al. (2008) y $PARAFAC$-PCA Wilderjans et al. (2009a).

El modelo $PARAFAC$-PCA (Wilderjans et al., 2009b) es un modelo para el análisis de un array de datos reales $(I \times J \times K)$ $\underline{\mathbf{D}}^1$ y una matriz de datos $(I \times L)$ $\mathbf{D}^2$ que tienen el primer modo en común. Se aproxima $\underline{\mathbf{D}}^1$ mediante una matriz del modelo $PARAFAC$ $\underline{\mathbf{M}}^1$ $(I \times J \times K)$ con P componentes y $\mathbf{D}^2$ se aproxima mediante el modelo PCA $\mathbf{M}^2$ con P componentes.

Es importante destacar que se restringe la matriz de componentes del primer modo para que sea la misma en ambos modelos. Por tanto podemos escribir $\underline{\mathbf{M}}^1$ y $\mathbf{M}^2$ como

sigue:

$$m^1_{ijk} = \sum_{p=1}^{P} a_{ip} b^1_{jp} c^1_{kp}$$

$$m^2_{il} = \sum_{p=1}^{P} a_{ip} b^2_{lp}$$

donde a_{ip}, b^1_{jp}, c^1_{kp} y b^2_{lp} son los elementos de las matrices de componentes $\mathbf{A}$ $(I \times P)$, $\mathbf{B}^1$ $(J \times P)$, $\mathbf{C}^1$ $(K \times P)$ y $\mathbf{B}^2$ $(L \times P)$.

El modelo combinado $HICLAS$-$INDCLAS$ para datos acoplados ($CHIC$) fue propuesto por Wilderjans et al. (2008). Aproxima un array de datos binarios $\underline{\mathbf{D}}^1$ (de tamaño $I \times J \times K$) y una matriz binaria de datos $\mathbf{D}^2$ (de tamaño $I \times L$) mediante un array binario del modelo $\underline{\mathbf{M}}^1$ (de tamaño $I \times J \times K$) y una matriz binaria del modelo $\mathbf{M}^2$ (de tamaño $I \times L$) de manera que: (1) $\underline{\mathbf{M}}^1$ y $\mathbf{M}^2$ se puedan descomponer mediante un modelo $INDCLAS$ y un modelo $HICLAS$ respectivamente de rango P (2) la matriz de paquetes objeto $\mathbf{A}$ es la misma en ambos modelos (sin pérdida de generalidad se ha considerado el modo objeto como modo común).

MÉTODO PROPUESTO: $T3$-PCA

El modelo propuesto, al que se denomina $T3$-PCA, aproxima un $I \times J \times K$ array de datos de tipo continuo $\underline{\mathbf{D}}^1$ que está acoplado con una $I \times L$ matriz de datos $\mathbf{D}^2$ a través del primer modo, mediante un array del modelo $\underline{\mathbf{M}}^1$ (de tamaño $I \times J \times K$) y una matriz del modelo $\mathbf{M}^2$ (de tamaño $I \times L$), respectivamente, tales que tienen el primer modo en común. En concreto, $\underline{\mathbf{D}}^1$ se aproxima mediante un $I \times J \times K$ array $\underline{\mathbf{M}}^1$ que se puede descomponer de acuerdo a un modelo $Tucker$ (Tucker, 1966; Kroonenberg and de Leeuw, 1980) con P componentes de objetos, Q componentes de atributos y R componentes de recursos; $\mathbf{D}^2$ se aproxima mediante una matriz $\mathbf{M}^2$ que se puede descomponer de acuerdo a un modelo PCA con P componentes, de manera que la matriz de componentes para el primer modo (i.e., las filas) en $\underline{\mathbf{M}}^1$ y $\mathbf{M}^2$ sea idéntica. Por tanto $\underline{\mathbf{M}}^1$ y $\mathbf{M}^2$ se pueden

descomponer como sigue:

$$m^1_{ijk} = \sum_{p=1}^{P}\sum_{q=1}^{Q}\sum_{r=1}^{R} g_{pqr}a_{ip}b^1_{jq}c_{kr}$$

$$m^2_{il} = \sum_{p=1}^{P} a_{ip}b^2_{lp}$$

donde a_{ip}, b^1_{jq}, c_{kr} y b^2_{lp} son los elementos de las matrices de componentes $\mathbf{A}(I \times P)$, $\mathbf{B}^1(J \times Q)$, $\mathbf{C}(K \times R)$ y $\mathbf{B}^2(L \times P)$, y P, Q y R denotan el número de componentes para objetos, atributos y recursos, respectivamente. Los elementos g_{pqr} son los elementos de la core matrix $\underline{\mathbf{G}}$ (de tamaño $P \times Q \times R$) y reflejan la relación entre la p^{th}, q^{th} y r^{th} componente de $\mathbf{A}$, $\mathbf{B}$ y $\mathbf{C}$, respectivamente. Señalar que $\underline{\mathbf{M}}^1$ y $\mathbf{M}^2$ estan relacionadas mediante la restricción de que matriz de componentes de objetos $\mathbf{A}$ sea igual en ambos submodelos.

El objetivo del análisis $T3$-PCA con (P, Q, R) componentes, de datos $(\underline{\mathbf{D}}^1, \mathbf{D}^2)$ es estimar un modelo $(\underline{\mathbf{M}}^1, \mathbf{M}^2)$ de manera que el valor de la función de pérdida:

$$f = \frac{\alpha}{\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}(d^1_{ijk})^2} \times \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}(d^1_{ijk} - \sum_{p=1}^{P}\sum_{q=1}^{Q}\sum_{r=1}^{R} a_{ip}b^1_{jq}c_{kr}g_{pqr})^2$$

$$+\frac{(1-\alpha)}{\sum_{i=1}^{I}\sum_{l=1}^{L}(d^2_{il})^2} \times \sum_{i=1}^{I}\sum_{l=1}^{L}(d^2_{il} - \sum_{p=1}^{P} a_{ip}b^2_{lp})^2, (0 \leq \alpha \leq 1)$$

sea mínimo, $\underline{\mathbf{M}}^1$ se pueda representar mediante un modelo $Tucker3$ con P, Q, R componentes, $\mathbf{M}^2$ por un modelo PCA con P componentes y la matriz de componentes $\mathbf{A}$ sea igual en la descomposición de $\underline{\mathbf{D}}^1$ y $\mathbf{D}^2$. Los pesos α y $1-\alpha$ denotan el peso con el que los datos de las matrices de tres y dos vías influyen en el análisis (Wilderjans et al., 2009a). Si $\alpha = 1$ entonces las componentes del modo común se determinan sólo en base a la matriz de tres vías (o lo que es lo mismo se aplica una estrategia segmentada que consiste en aplicar primero un $Tucker3$ y después un PCA), mientras que si $\alpha = 0$ sólo se considera la matriz de dos vías para la estimación de las componentes del modo común (o lo que es lo mismo se aplica una estrategia segmentada que consiste en aplicar primero un PCA y después un $Tucker3$).

Para estimar las matrices de componentes del modelo $T3$-PCA se utiliza un algorítmo basado en los mínimos cuadrados alternados (Kroonenberg and de Leeuw, 1980), algoritmo que se detalla en la tesis. Dado que los algoritmos ALS puede terminar en un mínimo local, la idea es ejecutar el algoritmo varias veces, cada vez con un valor inicial diferente, y retener el modelo para el cual la función de pérdida es menor.

Por tanto el objetivo es encontrar un modelo de rango (P, Q, R) de manera que exista un buen balance entre el ajuste del modelo (cuantificado por la función de pérdida) y la complejidad del modelo. Para determinar el rango del modelo $T3$-PCA utilizamos el procedimiento CHULL para la selección de modelos (Ceulemans and Kiers, 2009; Wilderjans et al., 2013). Para aplicar dicho método es necesario realizar análisis $T3$-PCA en rangos (1, 1, 1) hasta (P, Q, R) y calcular el correspondiente valor de la función de pérdida f y la complejidad del modelo (P + Q + R).

El modelo $T3$-PCA tiene libertad rotacional ya que el modelo $Tucker3$ también la tiene, por tanto se pueden rotar las matrices de componentes y la core matrix para buscar soluciones más facilmente interpretables.

Una vez presentado el modelo y el algoritmo que ajusta dicho modelo queremos evaluar el rendimiento de dicho algoritmo. En concreto estamos interesados en dos aspectos: optimización y recuperación de datos. En lo relativo a la optimización se trata de evaluar si el algoritmo es sensible a los mínimos locales. En cuento a la recuperación queremos determinar si el algoritmo recupera correctamente los datos y además evaluar si su rendimiento es superior al de otras estrategias. Más allá se investiga si estos dos aspectos dependen y de qué modo dependen de cinco características de los datos. Estas caracteristicas son: (1) el tamaño del bloque 3D, (2) la ratio entre el tamaño del bloque 3D y el bloque 2D, (3) el rango del modelo, (4) la importancia de una componente extra adicional y (5) la cantidad de ruido de los datos.

Los cinco factores anteriores se manipulan de la siguiente forma:

1. El tamaño del bloque de datos 3D ($I \times J \times K$), con dos niveles: ($20 \times 20 \times 20$) y ($20 \times \times 100$);

2. La ratio r entre el tamaño del bloque 3D y el tamaño del bloque 2D:

$$r = \frac{I \times J \times K}{I \times L} \tag{1}$$

   Este factor se manipula con los valores 1 y 0,25;

3. El rango del modelo ($P \times Q \times R$), con dos niveles ($2 \times 2 \times 2$), ($2 \times 3 \times 4$);

4. La cantidad de varianza explicada por una componente extra añadida (diferente para el bloque 3D y el bloque 2D). Este factor se manipula a tres niveles: 5 %, 15 % y 30 % y

5. La cantidad de ruido de los datos, ε, a 2 niveles: .15, .30

Cruzando todos los factores de diseño, 20 (replicaciones) x 2 (tamaño 3D) x 2 (ratio tamaño 2D / tamaño 3D) x 2 (rango) x 3 (tamaño de la componente adicional) x 2 (cantidad de ruido) x 10 (valor de alfa) tenemos 9.600 bloques de datos a analizar. Sobre estos bloques de datos se aplicarán los algoritmos $T3$-PCA y los algoritmos segmentados.

CONCLUSIONES

Entre los resultados más relevantes obtenidos tenemos los siguientes:

(1) El algoritmo $T3$-PCA es capaz de identificar el mínimo global de la función de pérdida. Sólo en 27 de los 9.600 conjuntos de datos el valor de la función de pérdida obtenida aplicando el algoritmo es mayor que el valor proxy. Además en todos estos casos el valor de alfa es 0,999.

(2) El mejor rendimiento del algoritmo $T3$-PCA se obtiene para alfa igual a 0,5. A medida que alfa se acerca a 1 el rendimiento del algoritmo empeora.

(3) El rendimiento del algoritmo $T3$-PCA en la recuperación de las matrices de componentes $\mathbf{A}$ y $\mathbf{B}^2$ disminuye a medida que variabilidad explicada por la componente extra adicional crece.

(4) El algoritmo $T3$-PCA funciona mejor que el algoritmo Segmentado3D en la recuperación de las matrices de componentes para valores de alfa próximos a 0,5.

(5) El algoritmo $T3$-PCA funciona mejor que el Segmentado3D en la recuperación de $\mathbf{A}$ cuando aumenta el porcentaje de variabilidad explicada por la componente extra.

PUNTOS ABIERTOS

Obviamente el modelo $T3$-PCA puede ser extendido y restringido de diferentes formas:

(1) Una posible mejora podría ser realizar una rotación combinada. Kiers (1998) desarrolló un procedimiento en el cual se puede dar un peso a cada matriz de componentes y a la core, este peso indica la importancia que la matriz/core tiene a la hora de la interpretación. Para el caso del modelo $T3$-PCA sería interesante desarrollar un procedimiento similar en el que el usuario pueda decidir el peso que asigna a las matrices de componentes $\mathbf{A}$, $\mathbf{B}^1$, $\mathbf{C}$, $\mathbf{B}^2$ y a la core $\underline{\mathbf{G}}$.

(2) Algunas posibles extensiones del modelo serían:

   a) Tener varios bloques de datos de dos vías acoplados a distintos modos;

   b) Tener múltiples bloques de dos vías para un modo;

   c) Imponer restricciones, por ejemplo no negatividad (cuando la matriz de tres vías se corresponde con niveles de emoción a lo largo del tiempo para diferentes personas, las componentes no deberían ser negativas).

INTRODUCCIÓN

En muchas ocasiones estamos interesados en describir las distintas interacciones que pueden estar presentes en tablas multivia de tipo continuo. Esto es lo que ocurre en el Análisis de Datos Genotipo x Ambiente, donde se mide el rendimiento de una serie de genotipos en diferentes ambientes. Puede o no existir replicación.

Para el caso de experimentos replicados, podemos utilizar técnicas clásicas como el Análisis de la Varianza para describir los efectos principales y determinar la existencia de interacción. Sin embargo no nos permite llegar mucho más allá.

Las técnicas estadísticas más utilizadas para el análisis de este tipo de datos son el modelo AMMI (Main effects and Multiplicative Interaction) (Gauch, 1988) y el modelo GGE (Genotype main effects and Genotype x Environment interaction effects) (Yan et al., 2000).

En los modelos AMMI se incorporan tantos téminos multiplicativos como sean necesarios para explicar la variabilidad de la interacción de segundo orden. Se basan en la descomposición en valores y vectores singulares (SVD) de la matriz de residuales de interacción del Modelo Lineal asociado.

El modelo GGE aplica la descomposición SVD a los datos a los que previamente se les ha sustraido la media de los ambientes. Por tanto se realiza la descomposición SVD a la matriz que combina los efectos principales genotipo más la interacción genotipo x ambiente.

En ambos modelos se utilizan las representaciones biplot propuestas por Gabriel (1971). El biplot es una representación gráfica que refleja en dimensión reducida las características más relevantes de una matriz de datos. La diferencia fundamental respecto a otras representaciones, es que en este caso se logra una representación conjunta; es decir, aparecen superpuestos en el mismo gráfico los puntos fila y los puntos columna; en nuestro caso los genotipos y los ambientes.

Actualmente la mayoría de los paquetes estadísticos incluyen algún procedimiento o macro para generar biplots. Uno de los programas comerciales existentes para el análisis biplot es el programa `GGEbiplot` (ver `www.ggebiplot.com`).

En la actualidad, es muy común el uso del paquete estadístico R entre la comunidad científica (RTe, 2011). R es un entorno informático estadístico que incluye herramientas de análisis de datos y de generación de gráficas. Es software libre y funciona bajo Windows, MAC OS y Linux (www.r-project.org).

Por todo ello parece interesante para la comunidad científica que exista un paquete en R que ofrezca una alternativa al software comercial existente hasta el momento. Este paquete de R deberá dar respuesta a todas las necesidades de los investigadores y además ser interactivo, para así ser más atractivo y no requerir para su uso de ningún conocimiento del lenguaje R.

Cuando hablamos de datos de tres vías nos referimos a datos que se almacenan en matrices tridimensionales. Podemos encontrarnos datos de este tipo en diversos contextos: medidas de ansiedad de diferentes individuos en diferentes situaciones, sintomas observados en diferentes pacientes por distintos médicos, imagenes de un PET-TAC tomadas en diferentes áreas del cerebro, medidas en distintos individuos realizando diferentes tareas mentales.

Podríamos pensar en analizar este tipo de datos colapsando una de las vías o bien analizando todos los conjuntos de datos de dos vías. Sin embargo estas opciones no ofrecen una descripción de la interacción existente entre los datos y nos pueden llevar a conclusiones incorrectas.

Las técnicas de análisis de tres vías resumen toda la información presente en los datos (i.e., todos los efectos principales y todas las interacciones), y además lo hacen de forma eficiente. En concreto, los métodos de análisis de tres vías resumen la información de cada modo por medio de unas cuantas componentes, y además describen las relaciones entre dichas componentes (Kiers and Mechelen, 2001).

Los dos métodos más importantes para datos de tipo continuo son el $PARAFAC$ (parallel factor analysis) propuesto por Harshman (1970) y Carroll and Chang (1970) y el“ Three-mode factor analysis“, propuesto por Tucker (1966) y posteriormente desarrollado por Kroonenberg and de Leeuw (1980). Estos últimos le dieron el nombre de “Three-mode principal component analysis” (3MPCA), recientemente a este método se

le conoce con el nombre de análisis $Tucker3$ (Kiers, 2000).

Para el análisis de datos de tipo binario se utilizan los modelos de clases jerárquicas ($HICLAS$). Estos constituyen la única familia de modelos en la que (a) se introduce simultaneamente una clasificación jerárquica (de los elementos) de cada modo y (b) se relacionan las clasificaciones jerárquicas por medio de una relación de asociación. En el caso de los arrays de datos de tres vías binarios, los modelos $HICLAS$ más importantes son los modelos $INDCLAS$ (Leenen et al., 1999) y $Tucker3$-$HICLAS$ (Ceulemans et al., 2003). El modelo $INDCLAS$ es el equivalente al modelo $PARAFAC$ para tres vías binarias, mientras que el modelo $Tucker3$-$HICLAS$ es el equivalente al modelo $Tucker3$.

La familia de modelos de clases jerárquicas $HICLAS$ constituye por tanto una familia de modelos para el análisis de datos binarios. Van Mechelen et al. (2007) proponen una extensión de dichos modelos para analizar tablas de datos de dos vías con datos tipo rating ($HICLAS$-R). El modelo $HICLAS$-R preserva las clasificaciones jerárquicas así como una generalización de las relaciones si-entonces presentes en los modelos $HICLAS$.

En la sociedad de la información en la que nos encontramos cada vez es más común el encontrarnos no sólo con volumenes de datos mayores y más cantidad de ellos, sino que también es común encontrarnos con conjuntos de datos que incluyen varios tipos de información sobre el mismo sistema. Estos conjuntos de datos se conocen con los términos *coupled*, *linked*, o *multiset data*, mientras que el análisis de datos asociado se conoce con el nombre de *data fusion*. Ejemplos de este tipo de datos podemos encontrarlos en psicometría (Wilderjans et al., 2009b,a), neuropsicología, bioinformática (Acar et al., 2012) y quimiometría (Smilde et al., 2000) entre otros. Para una revisión de los mismos ver Acar et al. (2013).

En el análisis de estos datos acoplados buscamos un submodelo para cada uno de los bloques de datos y una estructura que relacione dichos submodelos. En este modelo global el hecho de que los bloques compartan modos se refleja introduciendo constraints en las matrices de cuantificación de las modos compartidos (Van Mechelen and Smilde, 2010). Se pueden considerar una gran variedad de constraints pero la más simple de

ellas es la identidad. Esta restricción implica simplemente que los modos compartidos se cuantifican de igual forma en todos los submodelos donde aparecen.

Entre los modelos para el Análisis de Datos Acoplados más utilizados están el modelo $PARAFAC\text{-}PCA$ (Wilderjans et al., 2009a) y el modelo $CHIC$ (Wilderjans et al., 2008).

El modelo $PARAFAC\text{-}PCA$ permite el análisis integrado de un conjunto de datos (de tipo continuo) en tres vías y una tabla de dos vías que tienen un modo en común. El array de tres vías se ajusta según un modelo $PARAFAC$ mientras que la tabla de dos vías se ajusta según un modelo PCA (Pearson, 1901; Hotelling, 1933), con la restricción de que la matriz de componentes del modo común sea igual en ambos modelos.

El modelo $PARAFAC\text{-}PCA$ pertenece a la familia de modelos *multiway multiblock component models*, los cuales están muy relacionados con la familia de modelos *multiway covariate/multiblock regression models* (Smilde and Kiers, 1999; Smilde et al., 2000). Ambas familias de modelos sólo difieren en el rol que asignan a los diferentes bloques de datos. En el modelo *multiway covariate/multiblock regression models* los diferentes bloques de datos tienen distintos roles (i.e., un bloque sirve como bloque predictor mientras que el otro sirve como bloque criterio), mientras que el modelo *multiway multiblock component models* los dos bloques de datos son intercambiables en términos conceptuales.

$CHIC$ es un modelo para el análisis integrado de datos acoplados que consisten en una matriz binaria de dos vías y una matriz binaria de tres vías que tienen un modo en común. Este modelo contiene dos submodelos, ambos pertenecientes a la familia de modelos de clases jerarquicas para datos binarios de N vías (De Boeck and Rosenberg, 1988; Leenen et al., 1999; Ceulemans et al., 2003). Representa el array de datos binario de tres vías mediante un modelo $INDCLAS$ (Leenen et al., 1999) y el array de datos binario de dos vías mediante un modelo $HICLAS$ (De Boeck and Rosenberg, 1988) y relaciona ambos modelos imponiendo la restricción de que los cluster obtenidos para el modo común sean los mismos en ambos submodelos.

Este trabajo tiene como objetivo producir un marco metodológico general, incluyendo diferentes técnicas para el estudio de la interacción en matrices de dos y tres vías de tipo continuo y binario así como técnicas para el análisis de Datos Acoplados, además de desarrollar una nueva técnica para el Análisis de Datos Acoplados de tipo continuo que supone una mejora a las existentes hasta el momento.

El trabajo descrito en esta tesis persigue los siguientes objetivos:

(1) Hacer una revisión de los métodos para el análisis de la interacción en tablas de dos vías y desarrollar software libre para llevar a cabo dicho análisis.

(2) Hacer una revisión de los métodos para el análisis de la interacción en tablas de tres vías con datos de tipo real, centrándome fundamentalmente en los modelos $CANDECOMP/PARAFAC$ y $Tucker3$.

(3) Hacer una revisión de los métodos para el análisis de tablas de tres vías con datos de tipo binario, especialmente de los modelos $INDCLAS$ y $Tucker3$-$HICLAS$.

(4) Estudiar la aplicación de los métodos anteriores en el análisis de tablas de dos vías con datos tipo rating, revisión del modelo $HICLAS$-R.

(5) Estudio de estrategias existentes para el análisis de datos acoplados. Revisión de los modelos $PARAFAC$-PCA y $CHIC$.

(6) Proponer un nuevo modelo $T3$-PCA para el análisis de datos acoplados de tipo continuo. Además de presentarse el modelo se propondrá un algoritmo para el cálculo de los parámetros del modelo, un método para el cálculo del rango del mismo y rotaciones que mejoren la interpretabilidad de los resultados.

(7) Realizar un estudio de simulación que permita evaluar el rendimiento del algoritmo propuesto desde el punto de vista de la optimización del mismo y la capacidad para la recuperación de datos. Además se evaluará en qué casos el algoritmo $T3$-PCA recupera mejor la información que una estrategia segmentada.

Las técnicas para el análisis de datos que se han revisado y propuesto en esta tesis son aplicables a otras muchas áreas del conocimiento, como queda de manifiesto en la multitud de trabajos existentes en la literatura.

ESQUEMA DE CONTENIDOS

La parte central de esta tesis está organizada en siete capítulos.

El **CAPÍTULO 1** comienza con una revisión de los métodos biplot: biplot clásicos, HJ biplot y GGE biplot. Es al introducir el GGE biplot cuando se hace una revisión de los métodos existentes para el análisis de la interacción en tablas de dos vías, dentro del contexto de la evaluación de cultivos. Además se presenta una paquete desarrollado en el lenguaje `R` que implementa las principales funciones de los biplot clásicos y del GGE biplot. El capítulo finaliza con una aplicación a datos utilizando el paquete `GGEbiplotGUI` anteriormente mencionado.

Se incluye el paper **"An interactive biplot implementation in R for modeling genotype-by-environment interaction"** publicado en la Revista **STOCHASTIC ENVIRONMENTAL RESEARCH AND RISK ASSESSMENT** en el que se realiza una revisión de los métodos biplot en general y del GGE biplot en particular, además de presentarse el paquete `GGEbiplotGUI` junto con una aplicación a datos. Dicho paper se encuentra recogido en el capítulo **ARTÍCULOS**.

En el **CAPÍTULO 2** se realiza una exhaustiva revisión de los Modelos para Tablas de Tres Vías con datos continuos. Los modelos que se abordan con mayor detalle son los Modelos $CANDECOMP/PARAFAC$ y los Modelos $Tucker$. Para cada de uno de ellos se recoge su algoritmo correspondiente, los métodos para la elección del número de componentes que se han de retener y aplicaciones de dichos modelos.

En el **CAPÍTULO 3** se realiza una exhaustiva revisión de los Modelos para Tablas de Tres Vías con datos binarios. Para ello se introducen los Modelos de Clases Jerárquicas para Tablas de Dos Vías Binarias y a partir de estos se presentan los Modelos $INDCLAS$ y $Tucker3$-$HICLAS$ para el análisis de Tres Vías. Además se hace una pequeña aplicación a datos utilizando el modelo $Tucker3$-$HICLAS$.

En el **CAPÍTULO 4** se presenta el modelo $HICLAS$-R para el Análisis de Tablas de Dos Vías con datos tipo rating. Este modelo supone una extensión de los Modelos de Clases Jerárquicas introducidos en el Capítulo anterior y es una alternativa a la dicotomización que habitualmente se utiliza para el tratamiento de este tipo de

datos. Se presenta acompañado de una pequeña aplicación a datos que ilustra el mismo.

El **CAPÍTULO 5** se dedica al Análisis de Datos Acoplados presentándose la Familia de Modelos de Componentes Multivía-Multibloque. Se realiza un estudio de los modelos $PARAFAC$-PCA y $CHIC$ para el análisis de una tabla de tres vías y una de dos vías que tienen un modo en común.

En el **CAPITULO 6** se presenta un nuevo modelo, $T3$-PCA, que mediante una estrategia integrada permite el análisis de una matriz de tres vías y una matriz de dos vías, ambas con datos de tipo continuo, que tienen un modo en común. Más especificamente se presenta un modelo que consta de dos submodelos, un modelo $Tucker3$ para la matriz de tres vías y un modelo PCA para la matriz de dos vías. Estos dos modelos se relacionan imponiendo la restricción de que la matriz de componentes del modo común ha de ser igual en ambos submodelos. En este capítulo además de proponerse dicho modelo se presenta el algoritmo para el cálculo de los parámetros del modelo y un método para la selección del rango del mismo. Además se lleva a cabo un estudio de simulación para comparar el rendimiento de dicha estrategia con otras estragias (segmentadas y separadas). También se proponen algunas líneas de investigación futuras de dicho modelo.

Se incluye el paper **"Data fusion by $T3$-PCA: An integrated global model for the simultaneous analysis of coupled real-valued data"** sometido a la Revista **BEHAVIOR RESEARCH METHODS**. Dicho paper se encuentra recogido en el capítulo **ARTÍCULOS**.

En el **CAPÍTULO 7** se recoge el software que se ha desarrollado en `R` para poder llevar a cabo el estudio de simulación que se presenta en el capítulo anterior, además del análisis de datos según el algorítmo $T3$-PCA.

1

ANÁLISIS DE LA INTERACCIÓN EN TABLAS DE DOS VÍAS

1.1. INTRODUCCIÓN

Los métodos Biplot Clásicos introducidos por Gabriel (1971) , tienen como objetivo principal proporcionar una representación gráfica aproximada en baja dimensión de una matriz de datos rectangular (n individuos x p variables), con una calidad en la representación que permita captar visualmente gracias a sus propiedades geométricas, las interrelaciones entre conjuntos de individuos y variables, además de las relaciones entre los elementos de cada uno de esos conjuntos.

Se aproxima la matriz de datos a través de la Descomposición en Valores Singulares, para después realizar una factorización Biplot en marcadores fila y columna que permita la representación gráfica, cuya interpretación se basa en las propiedades geométricas del producto escalar entre vectores fila (denominados marcadores fila) y vectores columna (denominados marcadores columna), de tal forma que dicho producto reproduzca aproximadamente cada elemento en la matriz de datos (Golub and Reinsch, 1970).

Gower and Harding (1988), Gower (1992) and Gower and Hand (1996) proponen un enfoque diferente al propuesto por Gabriel (1971). Estos autores, obtienen primero una ordenación de los individuos utilizando métodos de escalamiento, para luego superponer en los biplots las variables, de forma tal que sea posible una interpretación conjunta. Este enfoque se puede relacionar, con la forma factorial clásica de la escuela francesa de análisis de datos y con los métodos de ordenación de la escuela biométrica.

Otro enfoque diferente al clásico es el que está basado en la relación entre la aproximación por mínimos cuadrados (LS) de las matrices y la descomposición SVD. Este tercer enfoque permite que los biplots sean vistos como modelos ajustados por bilinearidad. Estos modelos se conocen con el nombre de biplots de regresión y se pueden interpretar como modelos multiplicativos bilineales, permitiendo demostrar de forma gráfica la asociación, lineal o no, entre los sujetos y las variables (Gabriel, 1998; Vicente-Villardón et al., 2006). En esta línea, son muchos los autores que han utilizado los métodos biplot para explicar interacciones en tablas de dos vías (Denis, 1991; Falguerolles, 1995; Van Eeuwijk, 1995; Choulakian, 1966). Con el objetivo de aproximar estos modelos,

los autores anteriores utilizan los métodos de estimación usados en los modelos bilineales generalizados, los cuales son considerados como extensiones de los modelos lineales generalizados (Nelder and Wedderburn, 1972).

El biplot se ha convertido en una herramienta de visualización de datos muy popular en muchas áreas de investigación, tales como la psicología, la medicina, la sociología, la ecología y la agricultura. La primera aplicación del biplot en datos de agricultura la hicieron Bradu and Gabriel (1978) y fue para ilustrar el uso del biplot como herramienta de diagnóstico en la selección de modelos.

Otros trabajos que ilustran el uso del biplot en el análisis de datos genotipo x ambiente incluyen a Kempton (1984); Cooper and DeLacy (1994).

En los últimos años ha aparecido mucha literatura acerca del uso del Análisis Biplot en el estudio de la interacción genotipo x ambiente (GE). Este análisis biplot se realiza sobre uno de los diferentes modelos lineales-bilineales existentes. Los modelos más comunes en el análisis biplot de la interacción x genotipo ambiente han sido el modelo 'The additive main effects and multiplicative interaction' (AMMI) y el modelo 'Genotype main effects and genotype x environment interaction effects' (GGE) (Gauch, 1988).

El modelo AMMI ha sido aplicado en el análisis estadístico de ensayos de cultivos en multiples ambientes (MET); (Kempton, 1984; Gauch and Zobel, 1996, 1997; Ebdon and Gauch, 2002b,a). El modelo GGE fue propuesto por Yan et al. (2000). Existen muchas revisiones en las que se comparan y se contrastan los modelos AMMI y GGE para el análisis GE (Gauch, 2006; Yan and Tinker., 2006; Yan et al., 2007; Gauch et al., 2008). Los principios básicos de los mismos y su metodología han sido descritos también en libros sobre el AMMI (Gauch, 1992) y sobre el GGE (Yan and Kang, 2003).

En los modelos AMMI se introducen tantos términos multiplicativos como sean necesarios para explicar la interacción genotipo x ambiente. Estos modelos se basan en la Descomposición en Valores y Singulares (SVD) de la matriz de residuales del modelo. El modelo GGE aplica la descomposición SVD de los datos a los que se les sustraido previamente el efecto ambiente. De manera que el GGE biplot visualiza el efecto principal genotipo (G) y la interacción genotipo x ambiente (GE), ya que estas son las principales

fuentes de variación en la evaluación de cultivos (Kang, 1988; Gauch and Zobel, 1996; Yan and Kang, 2003). Este modelo se basa en el modelo 'Multiplicative linear-bilinear site regression' (SREG) (Cornelius et al., 1996).

La bibliografía sobre el biplot es muy extensa, existen multitud de publicaciones en la que se utiliza el biplot, y una busqueda en internet de la palabra 'biplot' nos devuelve alrededor de 164.000 resultados. Actualmente la mayoría de los paquetes estadísticos incluyen algún procedimiento o macro para generar biplots. Uno de los programas comerciales existentes para el análisis biplot es el programa `GGEbiplot` (ver `www.ggebiplot.com`).

En la actualidad, es muy común el uso del paquete estadístico `R` entre la comunidad científica (RTe, 2011). `R` es un entorno informático estadístico que incluye herramientas de análisis de datos y de generación de gráficas. Es software libre y funciona bajo Windows, MAC OS y Linux (`www.r-project.org`).

Por todo ello parece interesante para la comunidad científica que exista un paquete en `R` que ofrezca una alternativa al software comercial existente hasta el momento. Este paquete de `R` deberá dar respuesta a todas las necesidades de los investigadores y además ser interactivo, para así ser más atractivo y no requerir para su uso de ningún conocimiento del lenguaje `R`.

Este capítulo se organiza de la siguiente forma: en la sección 1.2 se hace una revisión del biplot y más concretamente del GGE biplot, en la sección 1.3 se presenta el paquete GGEBiplotGUI y en la sección 1.4 se aplica el paquete a un conjunto de datos reales.

En el Anexo **ARTÍCULOS**, se recoge el artículo publicado en la revista **Stochastic Environmental Research and Risk Assessment** (SERRA), en el que se desarrollan parte de los contenidos de este capítulo.

Frutos, E., Galindo, M.P., Leiva, V. (2014). An interactive biplot implementation in R for modeling genotype-by-environment interaction. Stochastic Environmental Research and Risk Assessment, 28(7):1629-1641.

Se incluye aqui un breve resumen del trabajo.

ABSTRACT

Classical and GGE biplot methods are graphical procedures that allow multivariate data to be analyzed. In particular, the GGE biplot displays the genotype main effect (G) and the genotype by environment interaction (GE) in two-way data. The GGE biplot originates from data graphical analysis of multi-environment variety trials (MET). Thus, agronomists, crop scientists and geneticists are potential users of this method. However, it can also be used to visualize and analyze other types of data. In this paper, we propose a new interactive computational implementation in `R` language to perform the main functions of the classical and GGE biplot methods, but it is also useful for MET data visual analysis. This implementation is organized in an `R` package named `GGEBiplotGUI`. This package is the only interactive, noncommercial and open source software that currently exists, offering a free alternative to existing commercial software. In addition, it can be used without to practically have knowledge of the `R` programming language. Here, we present and discuss the capabilities and features of the `GGEBiplotGUI` package and illustrate them by using real data. The `GGEBiplotGUI` package graphically addresses the questions that a researcher likely asks. This `R` package is not only a tool for visual analysis of MET data, but also a generic tool for visual analysis of other types of data. Although this package is mainly intended to the analysis of data generated by plant breeders and geneticists, in order to study yields from genotypes and interactions

between genotype and environment, data from other areas can also be analyzed by the `GGEBiplotGUI` package.

Keywords graphical display; least squares method; multivariate data analysis; singular value decomposition; statistical models; statistical software.

Mathematics subject classification Primary 62H99; Secondary 62-04.

1.2. REVISIÓN DEL BIPLOT

En este apartado vamos a dar un pequeño repaso a los métodos biplot clásicos y al GGE biplot.

1.2.1. BIPLOT CLÁSICOS

Cualquier $n \times m$ matriz $\mathbf{Y}$ de rango r se puede factorizar en una $n \times r$ matriz $\mathbf{G}$ y una $m \times r$ matriz $\mathbf{H}$, ambas de rango r, según la fórmula:

$$\mathbf{Y} = \mathbf{G}\mathbf{H}^{'}. \tag{1.1}$$

Esta factorización se puede escribir también como:

$$y_{ij} = g_i^{'} h_j, \tag{1.2}$$

para cada i y j, donde y_{ij} es el elemento que se encuentra en la fila i columna j de $\mathbf{Y}$, $g_i^{'}$ es la i-ésima fila de $\mathbf{G}$ y h_j es la j-ésima fila de $\mathbf{H}$.

La factorización de la Ecuación 1.1 asigna los vectores $g_1, \ldots, g_n$ (efectos fila) a cada una de las n filas de $\mathbf{Y}$, y los vectores $h_1, \ldots, h_m$ (efectos columna) a cada columna de $\mathbf{Y}$. Por tanto la Ecuación 1.1 proporciona una representación de $\mathbf{Y}$ por medio de $n+m$ vectores en el espacio r-dimensional. Cuando la matriz es de rango 2, estos $n+m$ vectores se pueden dibujar en el plano, dando una representación de los $n\,m$ elementos de $\mathbf{Y}$, por medio del producto interno de los correspondientes efectos fila y efectos columna. Este gráfico se conoce con el nombre de **biplot**, ya que permite que los efectos fila y columna se puedan representar de forma conjunta (Gabriel, 1971).

En resumen, un Biplot permite representar la fila i de la matriz de datos mediante el

marcador g_i y la columna j con el vector h_j, de forma que al proyectar el punto g_i sobre el vector h_j, esa proyección coincide con el valor y_{ij}. Esa es la traducción geométrica del concepto de producto escalar.

El interés práctico reside en que el orden de las proyecciones de cada marcador fila sobre un marcador columna reproduce el orden de la matriz de partida, de forma que analizando la posición de cada individuo sobre cada variable, es posible ordenar los individuos en función del valor que toman en esa variable, y eso puede hacerse para todos y cada uno de los variables (ver Figura 1.1).

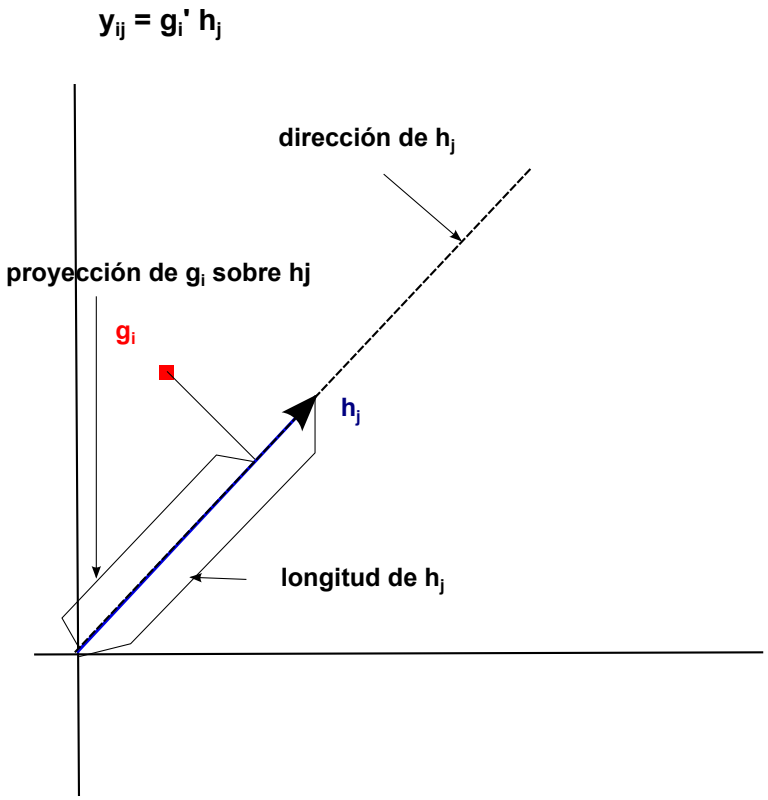


Figura 1.1: Representación geométrica del producto escalar.

Las matrices que tienen rango mayor que dos no se pueden representar de forma exacta mediante un biplot. Sin embargo, si una matriz $\mathbf{Y}$ se puede aproximar por una matriz de rango igual a dos ($\mathbf{Y}_{(2)}$), el producto interno de los efectos fila y los efectos

columna pueden aproximar a los elementos de $\mathbf{Y}$. Más concretamente, para aproximar cualquier matriz $n \times m$ $\mathbf{Y}$ de rango r por una matriz $n \times m$ de menor rango, se puede utilizar la descomposición SVD, que en este caso se escribiría como:

$$\mathbf{Y} = \mathbf{U}\Lambda\mathbf{V}' = \sum_{k=1}^{r} \lambda_k u_k v_k' \tag{1.3}$$

donde $\boldsymbol{U} = (\boldsymbol{u}_1, \ldots, \boldsymbol{u}_r)$, $\boldsymbol{V} = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_r)$ y $\boldsymbol{\Lambda} = (\lambda_1, \ldots, \lambda_r)$, los vectores $\boldsymbol{u}_k$ y $\boldsymbol{v}_k'$ se llaman vectores singulares izquierdos y vectores singulares derechos, respectivamente, y los valores λ_k son los valores singulares.

Para todo $k = 1, \ldots, r$, los valores singulares λ_k y los vectores singulares $\boldsymbol{u}_k$ y $\boldsymbol{v}_k$ se eligen de manera que satisfagan:

$$\begin{aligned} \boldsymbol{u}_k'\boldsymbol{Y} &= \lambda_k \boldsymbol{v}_k \\ \boldsymbol{Y}\boldsymbol{v}_k &= \lambda_k \boldsymbol{u}_k \\ \boldsymbol{Y}\boldsymbol{Y}'\boldsymbol{u}_k &= \lambda_k^2 \boldsymbol{u}_k \\ \boldsymbol{Y}'\boldsymbol{Y}\boldsymbol{v}_k &= \lambda_k^2 \boldsymbol{v}_k, \end{aligned}$$

con $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_r$, de manera que $\boldsymbol{u}_k'\boldsymbol{u}_l = \boldsymbol{v}_k'\boldsymbol{v}_l = \delta_{kl}$, donde δ_{kl} denota la delta de Kronecker.

Usando el método LS, la expresión dada en (1.3) se puede escribir como:

$$\boldsymbol{Y}_{(s)n\times m} = \boldsymbol{U}_{(s)n\times s}\boldsymbol{\Lambda}_{(s)s\times s}\boldsymbol{V}'_{(s)s\times m} = \sum_{k=1}^{s} \lambda_k \boldsymbol{u}_k \boldsymbol{v}_k', \tag{1.4}$$

que se corresponde con la mejor aproximación de $\boldsymbol{Y}$ de rango igual a s (Householder and Young, 1938), por tanto la expresión dada por la Ecuación (1.4) proporciona una matriz $n \times m$, llamémosla $\boldsymbol{M}$, de rango s que minimiza $\| \boldsymbol{Y} - \boldsymbol{M} \|^2 = \sum_{i=1}^{n} \sum_{j=1}^{m} (y_{ij} - m_{ij})^2$.

Dado que $\| \boldsymbol{Y} \|^2 = \lambda_1^2 + \cdots + \lambda_r^2$, la bondad de ajuste del modelo se puede medir mediante $\sum_{k=1}^{s} \lambda_k^2 / \sum_{k=1}^{r} \lambda_k^2 \times 100\,\%$.

También se pueden obtener las matrices de marcadores $\boldsymbol{G}$ y $\boldsymbol{H}$ por medio de regresiones alternadas. En concreto, si se consideran los marcadores fila $\boldsymbol{G}$ fijos, se pueden calcular los marcadores columna por regresión como (i) $\boldsymbol{H}' = (\boldsymbol{G}'\boldsymbol{G})^{-1}\boldsymbol{G}'\boldsymbol{Y}$. De la misma manera, fijando $\boldsymbol{H}$, se pueden obtener los marcadores fila como (ii) $\boldsymbol{G}' = (\boldsymbol{H}'\boldsymbol{H})^{-1}\boldsymbol{H}'\boldsymbol{Y}$. Por tanto alternado los pasos (i) y (ii) el resultado converge al SVD (Ukkelberg and Borgen, 1993).

Para elegir los factores $\boldsymbol{G}$, $\boldsymbol{H}$ de $\boldsymbol{Y}_{(2)}$, como aparecen en la ecuación (1.1), se puede utilizar la factorización obtenida por SVD tomando $\boldsymbol{G} = \boldsymbol{U}\boldsymbol{\Lambda}^{\gamma}$ y $\boldsymbol{H} = \boldsymbol{V}\boldsymbol{\Lambda}^{1-\gamma}$, donde el escalar γ puede tomar cualquier valor entre cero y uno.

Cuando $\gamma = 1$, los valores singulares se particionan completamente en los vectores propios fila, y el biplot obtenido se conoce con el nombre de **row-metric-preserving o JK biplot** (Gabriel, 1971). Dado que en este caso $\boldsymbol{G} = \boldsymbol{U}\boldsymbol{\Lambda}$, tenemos que $\boldsymbol{Y}\boldsymbol{Y}^{\top} = \boldsymbol{G}\boldsymbol{G}^{\top}$. Por tanto se preserva la métrica euclídea usual entre filas, y por tanto, es apropiado para evaluar similaridades/disimilaridades entre factores fila.

Cuando $\gamma = 0$, los valores singulares se particionan completamente en los vectores propios columna, y el biplot obtenido se conoce como **column-metric preserving o GH biplot** (Gabriel, 1971). Dado que ahora $\boldsymbol{H} = \boldsymbol{V}\boldsymbol{\Lambda}$, tenemos que $\boldsymbol{Y}'\boldsymbol{Y} = \boldsymbol{H}\boldsymbol{H}'$. Si $\boldsymbol{Y}$ está centrada, entonces $\boldsymbol{Y}'\boldsymbol{Y}$ es $(m-1)$ veces la matriz de covarianzas y el producto $(h_j'h_k)$ representa aproximadamente las covarianzas. Por tanto este particionamiento es apropiado para estudiar las relaciones entre los factores columna.

Dos reglas importantes en este particionamiento son: (i) la correlación entre dos columnas se aproxima por el coseno del ángulo que forman sus vectores (ii) la longitud del vector columna es igual a $\sqrt{m-1}$ veces la desviación estandar (SD) del factor columna a través de las filas.

Si $\gamma = 0{,}5$, se asignan roles simétricos a filas y columnas, y el biplot correspondiente se conoce con el nombre de **SQRT biplot o biplot simétrico**.

1.2.2. EL HJ BIPLOT

Galindo (1986) propone una nueva representación, conocida como **HJ biplot**, en la que las coordenadas para las columnas coinciden con los marcadores columna del GH biplot, mientras que las coordenadas para las filas coinciden con los marcadores fila del JK biplot. Estas coordenadas se pueden superponer en el mismo sistema de referencia con máxima calidad de representación.

El HJ biplot es similar al análisis de correspondencias (Galindo and Cuadras, 1986), pero no está restringido a datos de frecuencias. Las reglas para la interpretación del HJ biplot son una combinación de las reglas usadas en los biplots clásicos, el análisis de correspondencias, el análisis factorial y las técnicas de escalado multidimensional. En concreto, tenemos que:
(i) las distancias entre marcadores fila se interpretan como una función inversa de las similaridades, de manera que los marcadadores más cercanos (individuos) son más similares; esta propiedad permite identificar cluster de individuos con perfiles similares;
(ii) la longitud de los marcadores columna (vectores) aproximan la SD de las variables;
(iii) los cosenos de los ángulos entre los vectores columna aproximan las correlaciones entre variables de manera que ángulos agudos pequeños se asocian con variables con alta correlación positiva, ángulos obtusos indican correlación negativa y ángulos rectos señalan variables no correladas; de la misma forma, los cosenos de los ángulos entre los marcadores de las variables y los ejes (componentes principales) aproximan la correlación entre ambos. Para datos estandarizados, las cargas se aproximan a las de los factores en el Análisis Factorial; y
(iv) el orden de las proyecciones ortogonales de los marcadores fila (puntos) sobre un marcador columna (vector) aproxima el orden de los elementos fila (valores) en esa columna; por lo tanto la proyección de un punto (individuo) lejos del centro de gravedad (coordenada del punto medio), indica que este individuo toma en la variable valores mayores que la media.

1.2.3. EL GGE BIPLOT

En los últimos años se han logrado avances importantes en el uso de los modelos estadísticos multiplicativos para el análisis de ensayos de genotipos en ambientes múltiples y el estudio del complicado fenómeno de la interacción genotipo x ambiente (GEI). A estos modelos, Cornelius and Seyedsard (1997) los definen como modelos lineales-bilineales generales porque tienen términos aditivos (lineales) y términos multiplicativos (bilineales).

En general, los modelos multiplicativos tienen un componente aditivo (lineal) y un componente que consiste en la suma de terminos multiplicativos (bilineal).

Existen cinco modelos multiplicativos que pueden ser potencialmente usados en el análisis de g genotipos ($i = 1, 2, ..., g$) evaluados en e ambientes ($j = 1, 2, ..., e$).

El **modelo AMMI** (Additive Main effect and Multiplicative Interaction) (Gauch, 1988):

$$\overline{Y}_{ij} = \mu + \alpha_i + \beta_j + \sum_{k=1}^{t} \lambda_k \xi_{ik} \eta_{jk} + \overline{\varepsilon}_{ij}; \tag{1.5}$$

donde $\overline{Y}_{ij}$ es el promedio del i-ésimo genotipo en el j-ésimo ambiente, μ es la media global, α_i es el efecto genotipo, β_j es el efecto ambiente y $\overline{\varepsilon}_{ij}$ es el error aleatorio que se asume homogeneo con distribuciones normales e independientes $\overline{\varepsilon}_{ij} \sim N(0, \frac{\sigma^2}{n})$ (n = numero de repeticiones en cada ambiente; σ^2 = varianza del error aleatorio).

La GEI está dada por la suma de los términos multiplicativos $\sum_{k=1}^{t} \lambda_k \xi_{ik} \eta_{jk}$, donde t es el número de componentes retenidas en el modelo, λ_k es el valor singular para la componente k, los valores λ_k están ordenados $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_t \geq 0$, ξ_{ik} es el es el vector singular izquierdo de la k-esima componente y representa sensibilidades de los genotipos a ciertos factores ambientales hipotéticos dados por el vector singular derecho η_{jk} de la k-ésima componente.

Los ξ_{ik} y los η_{jk} están sujetos a restricciones de normalización $\sum_i \xi_{ik}^2 = \sum_j \eta_{jk}^2 = 1$ y

a restricciones de ortogonalidad $\sum_i \xi_{ik}\xi_{ik'} = \sum_j \eta_{jk}\eta_{jk'} = 0$ para $k \neq k'$.

En este modelo, los estimadores de mínimos cuadrados de los efectos aditivos, media general, efecto principal genotipo y efecto principal ambiente, no dependen de los estimadores de mínimos cuadrados de los efectos multiplicativos y son:

$$\begin{aligned}\widehat{\mu} &= \overline{y}_{..},\\ \widehat{\alpha}_i &= \overline{y}_{i.} - \overline{y}_{..},\\ \widehat{\beta}_j &= \overline{y}_{.j} - \overline{y}_{..}\end{aligned}$$

Los estimadores de mínimos cuadrados de los efectos multiplicativos son calculados de la descomposición en valores singulares de la matriz de residuales **(Z)**:

$$z_{ij} = \overline{y}_{ij} - \overline{y}_{i.} - \overline{y}_{.j} + \overline{y}_{..}$$

Del modelo AMMI se pueden suprimir los efectos principales de ambientes y así obtener el **modelo de regresión de genotipos (GREG)**:

$$\overline{y}_{ij} = \mu_i + \sum_{k=1}^{t} \lambda_k \xi_{ik}\eta_{jk} + \overline{\varepsilon}_{ij} \tag{1.6}$$

donde $\mu_i = \mu + \alpha_i$ Los estimadores de mínimos cuadrados de este modelo son:

$$\begin{aligned}\widehat{\mu}_i &= \overline{y}_{i.},\\ z_{ij} &= \overline{y}_{ij} - \overline{y}_{i.}\end{aligned}$$

El modelo GREG es una reparametrización del modelo convencional de estabilidad de Finlay and Wilkinson (1963) y Eberhart and Russell (1969) donde se hace la regresión del valor de los genotipos en los promedios o índices ambientales.

En el modelo GREG los vectores singulares de genotipos para el primer componente multiplicativo, ξ_{i1}, representan el coeficiente de regresión de cada genotipo, y los vectores

singulares de ambientes del primer componente multiplicativo, η_{j1}, serían los Indices ambientales. El valor λ_1 puede ser parcialmente absorbido en ξ_{i1} ó /y en η_{j1}. Todos los componentes multiplicativos mayores que el primero (t $>$ 1) se pueden combinar y se pueden considerar como la desviación de regresión.

Si se omiten del modelo AMMI los efectos principales de los genotipos, obtenemos el **modelo de regresión de ambientes (SREG)**

$$\overline{y}_{ij} = \mu_j + \sum_{k=1}^{t} \lambda_k \xi_{ik} \eta_{jk} + \overline{\varepsilon}_{ij} \tag{1.7}$$

donde $\mu_j = \mu + \beta_j$. Los estimadores de mínimos cuadrados de este modelo son:

$$\widehat{\mu}_j = \overline{y}_{.j},$$
$$z_{ij} = \overline{y}_{ij} - \overline{y}_{.j}$$

En este contexto se usa el término 'Genotypes Regression' en lugar de 'Rows Regression' de Bradu and Gabriel (1978). De igual manera, usamos el término 'Sites Regression' en lugar de 'Columns Regression'.

Si se suprimen tanto los efectos principales de genotipos como de ambientes del modelo AMMI, obtenemos el **modelo completamente multiplicativo** (COMM)

$$\overline{y}_{ij} = \sum_{k=1}^{t} \lambda_k \xi_{ik} \eta_{jk} + \overline{\varepsilon}_{ij} \tag{1.8}$$

donde el estimador de mínimos cuadrados del residual es $z_{ij} = \overline{y}_{ij}$.

El modelo COMM fue utilizado por Fisher and Mackenzie (1923).

El **modelo SHMM** tiene como efecto aditivo el parámetro β y es de la forma:

$$\overline{y}_{ij} = \beta + \sum_{k=1}^{t} \lambda_k \xi_{ik} \eta_{jk} + \overline{\varepsilon}_{ij} \tag{1.9}$$

Seyedsadr and L. (1992) han mostrado que este modelo es una reparametrización del modelo de no aditividad de Tukey (1949).

El modelo SHMM ha sido utilizado por Cornelius et al. (1992, 1993); Crossa et al. (1993, 1995) para identificar clusters de cultivos o de ambientes en los cuales los cambios en rangos de cultivos son estadisticamente insignificantes.

El ajuste de β por mínimos cuadrados requiere de un algoritmo iterativo, dado que $\widehat{z_{ij}} = \overline{y}_{ij} - \widehat{\beta}$ pero $\widehat{\beta} = \overline{y}_{..} - \sum_{k=1}^{t} \widehat{\lambda}_k \widehat{\overline{\alpha}}_k \widehat{\overline{\gamma}}_k$ (donde $\widehat{\overline{\alpha}}_k = \frac{\sum_i \widehat{\alpha}_{ik}}{g}, \widehat{\overline{\gamma}}_k = \frac{\sum_j \widehat{\gamma}_{jk}}{e}$) depende de la descomposición del valor singular de la matriz $\mathbf{Z}$, por lo que cambiando el numero de componentes multiplicativas cambia el valor de β.

El modelo **GGE** también se denomina a veces Sites Regression (SREG) model (Crossa and Cornelius., 1997). El GGE aplica la descomposición SVD a la matriz de datos que contiene el efecto genotipo (G) y el efecto interacción (GE), mientras que el modelo AMMI aplica la descomposición SVD a la matriz doble centrada, que contiene por tanto sólo el efecto interacción (GE).

Por tanto, el modelo GGE es:

$$Y_{ij} - \mu - \beta_j = \alpha_i + \phi_{ij} + \varepsilon_{ij}.$$

De manera que el método GGE mezcla G y GE en t términos multiplicativos:

$$Y_{ij} - \mu - \beta_j = \sum_{k=1}^{k} \lambda_k \xi_{ik} \eta_{jk}.$$

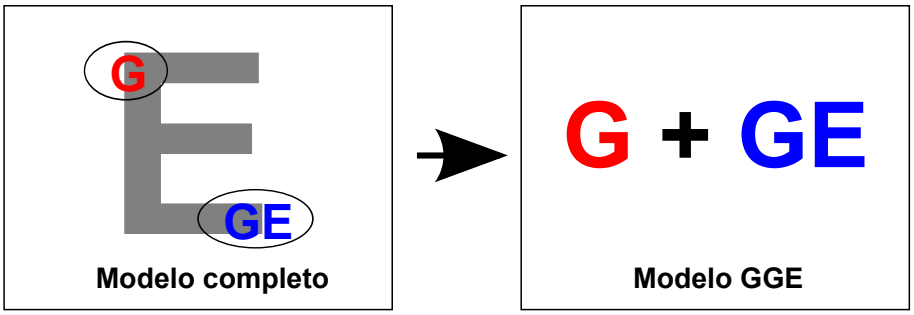


Figura 1.2: Comparación del modelo completo y el modelo GGE.

Para distinguir los miembros de una familia de modelos, se considera el número de componentes que se tienen en cuenta. Por ejemplo, el modelo GGE que tiene dos componentes se denomina GGE2 y el modelo AMMI con dos componentes se llama AMMI2. Los biplot más comunes en la literatura sobre evaluación de cultivos son dos: (i) El GGE2 biplot que tiene la primera componente (PC1) por abcisa y la segunda componente (PC2) como ordenada; y (ii) El AMMI2 biplot que tiene la primera componente (PC1) por abcisa y la segunda componente (PC2) como ordenada.

La interpretación del GGE2 y el AMMI2 es similar. Los genotipos más similares están más próximos en el gráfico que aquellos que no lo son. Lo mismo es cierto para los ambientes. Genotipos/ ambientes que son similares tienden a agruparse juntos. El ángulo entre los vectores ambientales están relacionados con la correlación entre los ambientes. Un ángulo agudo indica correlación positiva, mientras que un ángulo recto indica no correlación y un ángulo obtuso indica correlación negativa. La proyección de un genotipo en un ambiente indica el rendimiento del genotipo en dicho ambiente.

1.3. EL PAQUETE GGEBiplotGUI

En este apartado, se van a describir las distintas funciones del paquete `GGEBiplotGUI`. Este paquete incorpora los biplots tridimensionales gracias al paquete `rgl`.

Para la utilización del paquete `GGEBiplotGUI` no es necesario que el usuario tenga conocimiento del lenguaje de programación `R`.

Las funciones del paquete se organizan en las distintas opciones de menú:

File ⇒ **View** ⇒ **Biplot tools** ⇒ **Format** ⇒ **Models** ⇒ **Biplot**.

La mayoría de estas funciones no requieren de explicación alguna, por lo que sólo se van a describir aquellas que puedan plantear alguna duda.

En primer lugar se deberá descargar e instalar el paquete `GGEBiplotGUI` desde la página www.r-project.org, este paquete depende de los paquetes `rgl` y `tkrplot`, por tanto estos paquetes deberán instalarse también. A continuación se deberá cargar el paquete mediante el comando:

```
library("GGEBiplotGUI")
```

Una vez que todo lo anterior se ha llevado a cabo, se deberán cargar los datos, supongamos que sean `data1`. Para ejecutar el programa sobre estos datos se introducirá el comando:

```
GGEBiplot(data1)
```

Una vez introducida la sentencia anterior, aparecerá una ventana emergente con el nombre de "Model selection", desde la que se podrá seleccionar el modelo con el que se va a trabajar eligiendo el tipo de SVD que se quiere realizar:

1. "JK" (row metric preserving);

2. "GH" (column metric preserving);

3. “HJ” (dual metric preserving); y

4. “SQ” (symmetrical);

el tipo de centrado con::

1. “0” (no centering);

2. “1” (global centering: E + G + GE);

3. “2” (tester centering: G + GE); y

4. “3” (double centering: GE);

y, finalmente, el tipo de escalado entre:

1. “0” (no scaling); y

2. “1” (SD).

Una vez seleccionado el modelo, se pincha en el botón OK y a continuación se muestra el GGE biplot de `data1` en una ventana interactiva. En esta ventana aparecerá un menu con las siguientes opciones:

File

- Open log file.
- Copy image: copia la imagen al portapapeles.
- Save image: guarda la imagen en formato `pdf`, `postscript`, `bmp`, `png`, `jpg`/`jpeg` or `teX`.
- Print image: imprime la imagen en cualquiera de las impresoras conectadas al ordenador.
- Exit: finaliza la ejecución del programa.

View

- Show both: muestra genotipos y ambientes.
- Show genotypes: muestra sólo los genotipos.
- Show environments: muestra sólo los ambientes.
- Show/hide title.
- Show/hide guidelines.

Biplot tools

- Examine a genotype: Se selecciona un genotipo y se muestra el rendimiento del mismo en los distintos ambientes.
- Examine an environment: Se selecciona un ambiente y se muestra el rendimiento de los distintos genotipos en el ambiente seleccionado.
- Relation among environments.
- Compare two genotypes: Permite la comparación visual de dos genotipos en los distintos ambientes.
- Which won where/what: Permite identificar el mejor genotipo en cada ambiente.
- Discriminativeness against representativeness: Se define un ambiente promedio y se dibuja el eje ambiente promedio (AEA). Con ello se puede estudiar la representatividad y el poder discriminante de los ambientes de prueba.
- Mean against stability: Se calcula el ambiente promedio y se dibuja el AEA. Con todo ello se puede evaluar el rendimiento medio y la estabilidad de los cultivos.
- Rank environments with ref. to the ideal environment: Se define un ambiente ideal y se comparan todos los ambientes con respecto a el. El ambiente ideal de prueba se define como el más discriminante y el absolutamente representativo. Ello nos va a permitir ordenar los ambientes de prueba con respecto a ambos criterios.
- Rank genotypes with ref. to the ideal genotype: Se define un genotipo ideal y se comparan todos los genotipos respecto a el. El genotipo ideal se define como aquel que tiene el rendimiento más alto en todos los ambientes y además es absolutamente estable. Ello nos va a permitir ordenar los cultivos en terminos de rendimiento y estabilidad.

- Back to original data: Vuelve a dibujar el biplot original (previo a los cambios).

Format

- Biplot title: Permite cambiar el título del biplot.
- Change color con las opciones: color de fondo, etiquetas de genotipos, etiquetas de ambientes, y título del biplot.
- Change font con las opciones: por defecto, grande, y pequeña.

Models

Esta opción nos permite modificar la selección realizada anteriormente en la pantalla "Model selection":

- Scaled (divided by):
  - No scaling.
  - SD.
- Centered by:
  - no centering.
  - global-centered.
  - tester-centered.
  - double-centered.
- SVP:
  - JK (el método SVP utilizado es "row metric preserving").
  - GH (el método SVP utilizado es "column metric preserving").
  - HJ (el método SVP utilizado es "dual metric preserving").
  - SQ (el método SVP utilizado es "symmetrical").

Biplot

- PC1 against PC2 (default):
  Muestra la primera componente principal (PC1) frente a la segunda componente principal (PC2).
- PC3 against PC4:
  Muestra la tercera componente principal (PC3) frente a la cuarta componente principal (PC4).
- PC5 against PC6:
  Muestra la quinta componente principal (PC5) frente a la sexta componente principal (PC6).
- PC1 against PC3:
  Muestra la primera componente principal (PC1) frente a la tercera componente principal (PC3).
- PC2 against PC3:
  Muestra la segunda componente principal (PC2) frente a la tercera componente principal (PC3).
- Biplot3D:
  Muestra el biplot tridimensional.

1.4. ANÁLISIS DE DATOS

En esta sección, vamos a explicar el funcionamiento del paquete `GGEBiplotGUI` mediante su aplicación a una serie de datos. Estos datos son relativos a la evaluación del rendimiento del trigo de invierno (*Triticum aestivum* L.) en Ontario 1993, se evaluan 18 cultivos en 9 localidades (Yan and Kang, 2003). Los datos, a los que nos vamos a referir como 'datos de ontario' o simplemente `ontario`, se muestran en la Tabla 1.1.

Genotypes	BH93	EA93	HW93	ID93	KE93	NN93	OA93	RN93	WP93
Ann	4.46	4.15	2.85	3.08	5.94	4.45	4.35	4.04	2.67
Ari	4.42	4.77	2.91	3.51	5.70	5.15	4.96	4.39	2.94
Aug	4.67	4.58	3.10	3.46	6.07	5.03	4.73	3.90	2.62
Cas	4.73	4.75	3.38	3.90	6.22	5.34	4.23	4.89	3.45
Del	4.39	4.60	3.51	3.85	5.77	5.42	5.15	4.10	2.83
Dia	5.18	4.48	2.99	3.77	6.58	5.05	3.99	4.27	2.78
Ena	3.38	4.18	2.74	3.16	5.34	4.27	4.16	4.06	2.03
Fun	4.85	4.66	4.43	3.95	5.54	5.83	4.17	5.06	3.57
Ham	5.04	4.74	3.51	3.44	5.96	4.86	4.98	4.51	2.86
Har	5.20	4.66	3.60	3.76	5.94	5.35	3.90	4.45	3.30
Kar	4.29	4.53	2.76	3.42	6.14	5.25	4.86	4.14	3.15
Kat	3.15	3.04	2.39	2.35	4.23	4.26	3.38	4.07	2.10
Luc	4.10	3.88	2.30	3.72	4.56	5.15	2.60	4.96	2.89
M12	3.34	3.85	2.42	2.78	4.63	5.09	3.28	3.92	2.56
Reb	4.38	4.70	3.66	3.59	6.19	5.14	3.93	4.21	2.93
Ron	4.94	4.70	2.95	3.90	6.06	5.33	4.30	4.30	3.03
Rub	3.79	4.97	3.38	3.35	4.77	5.30	4.32	4.86	3.38
Zav	4.24	4.65	3.61	3.91	6.64	4.83	5.01	4.36	3.11

Tabla 1.1: Rendimiento medio de 18 variedades de trigo de invierno en 9 localidades de Ontario (datos de Ontario).

Los datos se deben cargar usando `R` Commander ó bien se pueden importar en `R` y

después guardarse como matriz o data frame. Los datos de Ontario están incluidos en el paquete como un data frame.

Para ejecutar el paquete con los datos de `ontario`; se debe escribir:

```
GGEBiplot(Data = Ontario)
```

y pulsar intro. Aparecerá la ventana "Model Selection‘" (Figura 1.3)

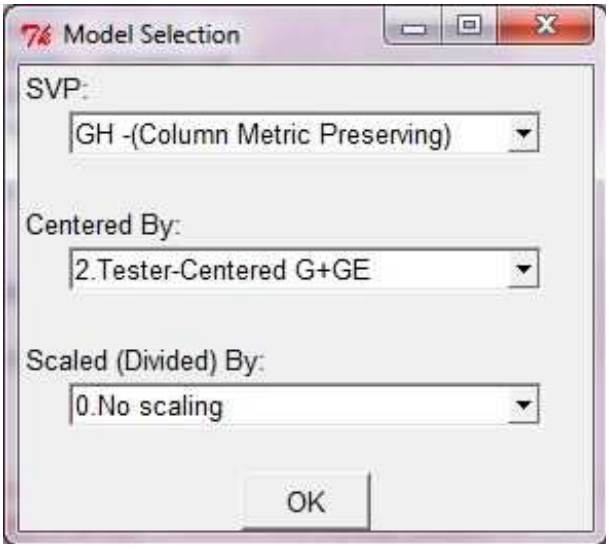


Figura 1.3: Pantalla para la selección del modelo.

Una vez seleccionado el modelo, pulsaremos el botón OK y aparecerá la ventana principal (Figura 1.4) del paquete donde se muestra un GGE biplot. En el GGE biplot aparecerán los genotipos en letra minúscula y color verde mientras que los ambientes aparecen en azul y mayúsculas. También aparece en el gráfico el porcentaje de variabilidad explicado por cada uno de los ejes.

El paquete `GGEBiplotGUI` dispone de diversas opciones para visualizar el GGE biplot, lo cual nos permitirá dar respuesta a las distintas preguntas que puedan hacerse los evaluadores de cultivos. A continuación se irán detallando cada una de estas opciones.

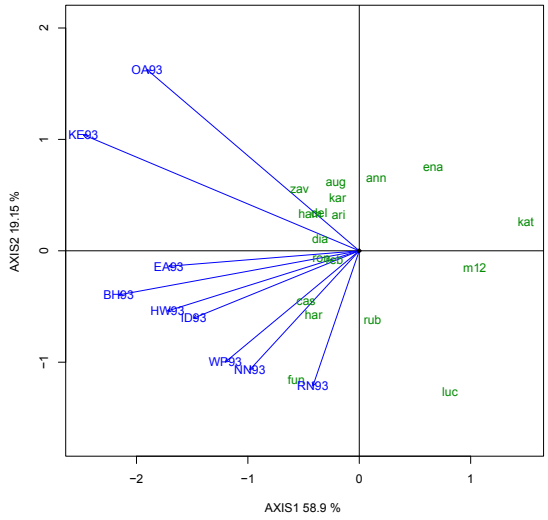


Figura 1.4: Pantalla principal (datos de Ontario).

1.4.1. RENDIMIENTO DE LOS DISTINTOS CULTIVOS EN UN DETERMINADO AMBIENTE

Si fuéramos productores de cultivos podríamos estar interesados en conocer qué variedades son las que más se adaptan a nuestro ambiente. El paquete `GGEBiplotGUI` da respuesta a esta pregunta mediante la opción de menú "Examine an environment" que se encuentra dentro de la barra de menú **Biplot tools**.

Se elige el ambiente que se desea evaluar (de entre los que se muestran en un combobox) y se acepta la selección. En nuestro caso seleccionamos BH93 y aparece la Figura 1.5 (izquierda), donde podemos ver:

- Una línea que pasa por el origen del biplot y el ambiente seleccionado (BH93, en

este caso), a la que denominamos eje ambiente.

- Una línea que pasa por el origen del biplot y que es perpendicular al eje ambiente.
- Las proyecciones de los cultivos sobre el eje ambiente.

Los cultivos se ordenan en la dirección del eje ambiente, de manera que en nuestro ejemplo el cultivo 'fun' es el mejor, seguido por 'cas' y 'har', mientras que 'kat' es el peor en el ambiente de prueba BH93.

La línea perpendicular separa los cultivos cuyo rendimiento está por encima de la media de aquellos que con rendimiento inferior a la media. Así, los cultivos 'kat', 'm12', 'ena', 'luc', and 'ann' tienen rendimiento por debajo de la media, mientras que el resto de los cultivos tienen rendimiento por encima de la media.

La longitud del vector que une el origen del biplot con el marcador ambiente es una medida de la capacidad del medio ambiente para discriminar entre cultivos. Un vector corto, en relación con el tamaño biplot, implica que todas las variedades tienden a tener similar rendimiento en el medio ambiente asociado.

1.4.2. ADAPTACIÓN DE UN CULTIVO A LOS DISTINTOS AMBIENTES

Los evaluadores de cultivos también pueden estar interesados en conocer qué entorno es el más adecuado para un determinado cultivo. Para dar respuesta a esta pregunta podemos utilizar la opción de menú "Examine a genotype" que se encuentra dentro de la barra de menú **Biplot tools**. Se elige el genotipo que se quiere evaluar (en nuestro ejemplo, fun) de entre los que aparecen en el combo-box, y se acepta la selección. Aparecerá la Figura 1.5(derecha) donde podemos ver:

- Una línea que pasa por el origen del biplot y el cultivo seleccionado, a la que vamos a llamar eje genotipo.
- Un línea que pasa por el origen del biplot y es perpendicular al eje genotipo.
- Las proyecciones de los ambientes sobre el eje genotipo.

Los ambientes se ordenan a lo largo del eje fun en la dirección indicada por la flecha. Por tanto el rendimiento de fun en los diferentes ambientes fue $WP93 \cong NN93 \cong BH93 \cong RN93 \cong ID93 \cong HW93 > EA93 > KE93 > OA93$. La línea que pasa por el origen del biplot y que es perpendicular al eje fun separa los ambientes en los cuales el rendimiento de fun está por debajo de la media, es decir OA93 de aquellos en los cuales el rendimiento está por encima de la media, es decir los restantes.

La distancia entre el origen del biplot y el marcador de un cultivo se llama vector cultivo. La longitud de este vector es una medida de la capacidad de respuesta del cultivo al medio ambiente. Un vector corto en relación al tamaño del biplot implica que la variedad asociada tiende a ser estable en todos los ambientes.

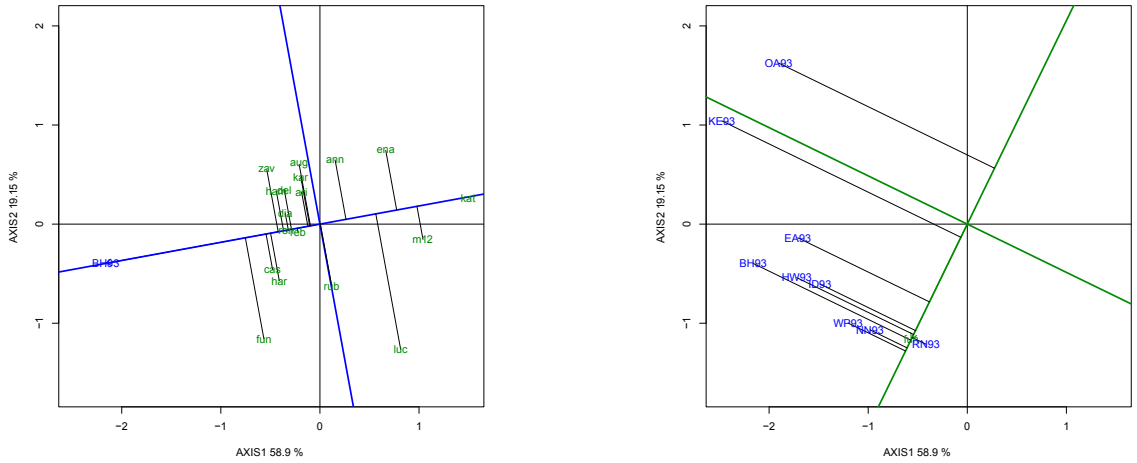


Figura 1.5: Ranking de cultivos basado en su rendimiento en el ambiente BH93 (izquierda) y ranking de ambientes basado en el rendimiento del cultivo fun (derecha).

1.4.3. COMPARACIÓN DE DOS GENOTIPOS

Desde la barra de menú Biplot tools, se selecciona la opción "Compare two genotypes". Se deberán elegir los dos cultivos que se quieren comparar y a continuación aceptar la selección, en nuestro ejemplo se seleccionan los cultivos fun y zav. Se mostrará una ventana similar a la Figura 1.6 izquierda, conteniendo los siguientes elementos:

- Dos óvalos que encierran a los cultivos seleccionados.
- Una línea que conecta los dos cultivos.
- Una línea perpendicular a la anterior y que pasa por el origen del biplot.

Existen dos ambientes OA93 y KE93 que están al mismo lado de la perpendicular que zav y los otros siete ambientes están al otro lado de la perpendicular, junto con fun. Esto indica que el rendimiento de zav es mayor que el de fun en OA93 y KE93, pero es

menor que el de fun en los otros siete ambientes (Yan and Tinker, 2006).

Esta interpretación se basa en la propiedad del producto interno. Basándonos en esta propiedad los cultivos fun y zav deberían tener el mismo rendimiento en cualquier ambiente que está localizado sobre la línea perpendicular, porque sus proyecciones sobre la misma son iguales. El cultivo zav tiene una proyección más larga sobre los vectores ambiente que están de su lado con respecto a la perpendicular y una proyección más corta sobre los vectores ambiente que están al otro lado respecto a la perpendicular. Una proyección más larga significa mayor rendimiento y viceversa.

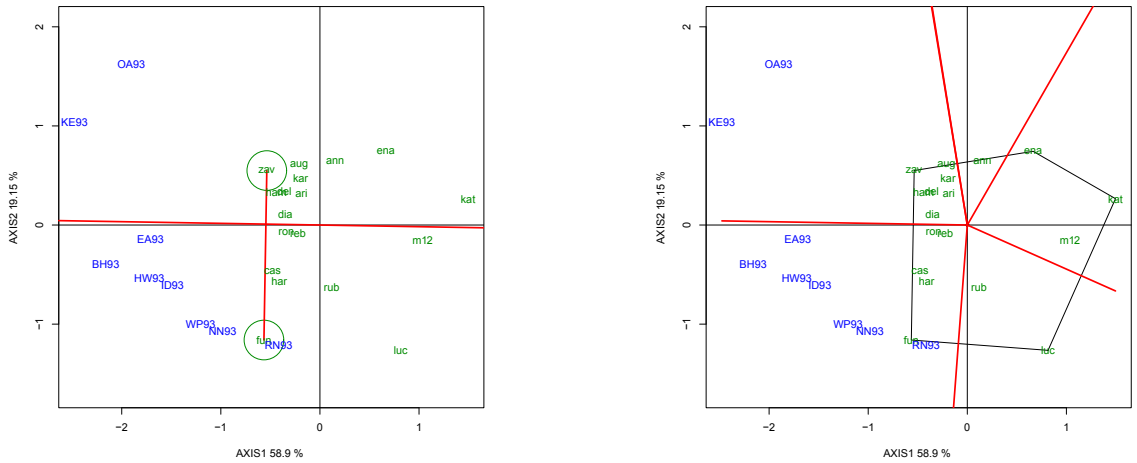


Figura 1.6: Comparación de los cultivos fun y zav (izquierda) y patrón which-won-where (derecha).

1.4.4. WHICH-WON-WHERE

Desde la barra de menú Biplot Tools se selecciona la opción "Which won where/what". El GGE biplot se presentará entonces como en la Figura 1.6 (derecha).

En ella podemos ver un polígono que se construye uniendo los cultivos que se encuentran más alejados del origen del biplot (fun, zav, ena, kat y luc), de manera que todos los cultivos se encuentran dentro del polígono. Este polígono se conoce como convex hull y a los cultivos situados en los vértices se les conoce como cultivos vértice. Estos cultivos por ser los mas alejados del origen serán por tanto los más sensibles. Además, aparecen una serie de líneas perpendiculares a cada uno de los lados del polígono. Estas perpendiculares dividen al biplot en varios sectores.

Hay cinco sectores con cultivos vértice fun, zav, ena, kat y luc. Los ambientes OA93 y KE93 están en el sector en el cual 'zav' es el cultivo vértice. Esto significa que zav es el mejor cultivo para OA93 y KE93. Los otros siete ambientes caen el sector donde fun es el cultivo vértice, lo cual significa que fun es el mejor cultivo en estos siete ambientes. No hay ningún ambiente en los sectores con vértices luc, ena y kat, lo cual indica que estos cultivos no son los mejores en ningún ambiente.

1.4.5. RENDIMIENTO MEDIO Y ESTABILIDAD DE LOS GENOTIPOS

Desde la barra de menú Biplot Tools, se selecciona la opción "Mean against stability" (Figura 1.7(izquierda)).

Para evaluar el rendimiento medio y la estabilidad de los genotipos se dibuja el AEC (average environment coordinate) sobre el JK biplot. En primer lugar se representa el ambiente promedio que se obtiene como la media de las coordenadas de los ambientes (se representa mediante un pequeño círculo).

La recta que pasa por el origen del biplot y el ambiente promedio se denomina eje de abscisas del ambiente promedio (abscisa del AEC). Las proyecciones de los marcadores genotipo sobre este eje deben por lo tanto aproximar los rendimientos medios de los genotipos y por tanto los cultivos aparecerán ordenados a lo largo del eje con la flecha apuntando al mayor rendimiento.El cultivo fun es el cultivo con mayor rendimiento medio en el mega-ambiente, seguido por cas, har, etc. La ordenada del AEC es la recta que pasa por el origen del biplot y es perpendicular al eje de abscisas anterior. La ordenada del AEC aproxima la GEI asociada a cada genotipo y por tanto mide la variabilidad o inestabilidad de los genotipos. Mayor proyección sobre el eje de ordenadas signica mayor inestabilidad. Por lo tanto rub y dia son más variables y menos estables que los otros cultivos. Los cultivos más cercanos al eje de abscisas son los más estables: cas, zav, reb, del, ari y kar.

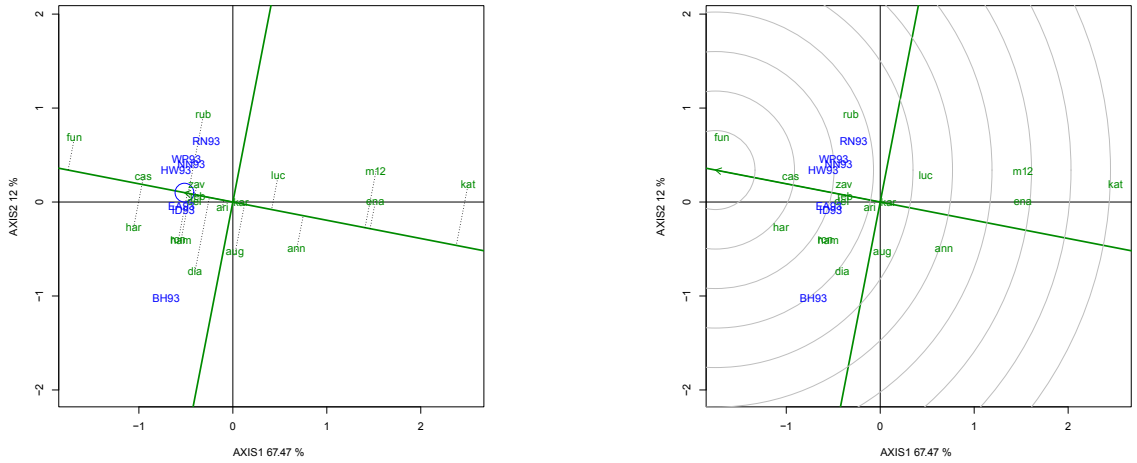


Figura 1.7: Rendimiento medio y estabilidad de los genotipos (izquierda) y ranking de genotipos respecto al genotipo ideal (derecha).

1.4.6. ORDENACIÓN DE GENOTIPOS RESPECTO AL GENOTIPO IDEAL

El genotipo ideal debe tener al mismo tiempo el mayor rendimiento medio y alta estabilidad en todos los ambientes. Por tanto el genotipo ideal es un punto del AEA (absolutamente estable) en la dirección positiva y la longitud de su vector asociado debe ser igual a la del mayor vector genotipo del lado positivo del AEA (mayor rendimiento medio). El genotipo ideal normalmente no existe, pero a pesar de ello se puede utilizar como referencia en la evaluación de cultivos. Se puede utilizar la distancia en el gráfico entre cualquier genotipo y el genotipo ideal como una medida de su deseabilidad. Los círculos concéntricos con centro el genotipo ideal permiten visualizar la distancia entre todos los genotipos y el genotipo ideal (Figura 1.7 derecha). Por tanto, 'fun' es el más

deseable mientras que 'cas' es el peor genotipo.

1.4.7. REPRESENTATIVIDAD Y PODER DISCRIMINANTE DE LOS AMBIENTES DE PRUEBA

La longitud del vector, es decir, la distancia entre el marcador del ambiente y el origen del biplot, es una medida de su poder discriminante: cuanto mayor sea su longitud mayor es su poder discriminante. Por tanto, de los nueve ambientes estudiados, KE93 y OA93 son los más discriminantes y RN93 es el menos discriminante, según puede verse en la Figura 1.8 izquierda. El vector ambiente promedio (representado por un pequeño circulo al final de la flecha) tiene las coordenadas promedio de todos los ambientes de prueba. El AEA es la linea que pasa por el ambiente promedio y el origen del biplot. Un ambiente de prueba que forma un ángulo pequeño con el AEA es representativo. Por tanto, BH93 es el más representativo mientras que OA93 es el menos representativo.

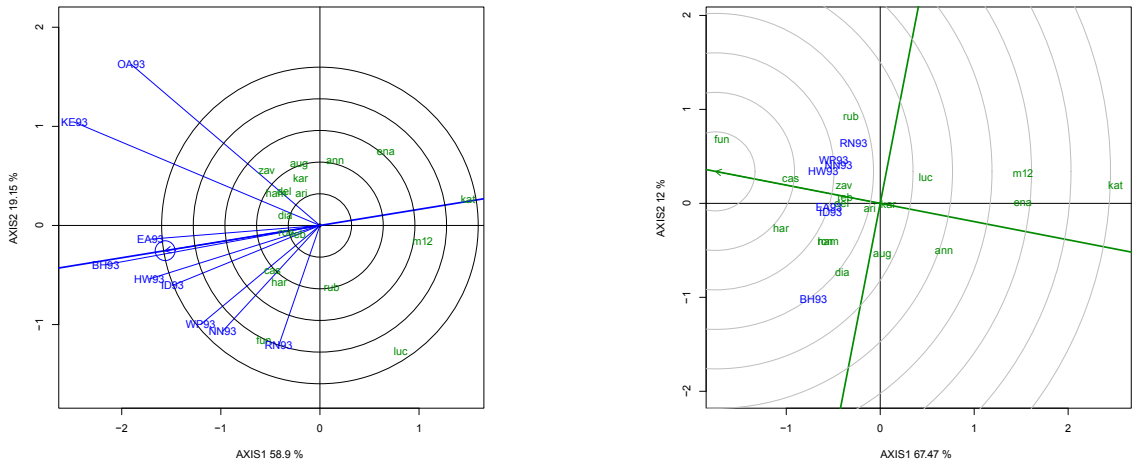


Figura 1.8: Representatividad y poder discriminante de los ambientes de prueba (izquierda) y ranking de ambientes (derecha).

1.4.8. ORDENACIÓN DE AMBIENTES

El ambiente ideal de prueba ha de ser el más discriminante y el más representativo de entre todos los ambientes de prueba. En la Figura 1.8 (derecha) se define el ambiente de prueba ideal, este es el centro de los circulos concéntricos. Es un punto en el eje AEA (el más representativo) con distancia al origen del biplot igual a la del vector ambiente más largo (más discriminante). BH93 es el más cercano a este punto y por tanto es el mejor, mientras que KE93 y OA93 son los peores para la selección de cultivos más adaptados a la región.

2

ANÁLISIS DE DATOS DE TIPO CONTINUO, EN TRES VÍAS

2.1. INTRODUCCIÓN

En el análisis de datos de tres vías la información se presenta en arreglos donde cada dato se indexa con tres índices: uno que indentifica al individuo i $(i = 1, ..., I)$, otro a la variable j $(j = 1, ..., J)$ y un tercero que corresponde a la condición k $(k = 1, ..., K)$; representándose en un arreglo tridimensional: individuos, variables y condiciones.

Los métodos utilizados para el ánalisis de datos de tres vías son de carácter exploratorio, porque identifican patrones de la estructura interna presente entre los elementos de las tres vías del arreglo, sin aplicar pruebas a estos patrones porque probar hipótesis supone un supuesto distribucional que en muchas ocasiones se desconoce (Kroonenberg, 2008).

Existen diferentes métodos de ánalisis provenientes de escuelas como la anglosajona y la francesa. Estos métodos se pueden clasificar en **asimétricos** y **simétricos**. Los asimétricos se denominan así porque uno de los modos (generalmente las ocasiones) tiene un tratamiento diferente a los otros (individuos y variables), hecho que no se da en los métodos simétricos donde se tratan por igual los tres modos. Además, los asimétricos están dentro del denominado enfoque *Interestructura-Compromiso-Intraestructura* (ICI), nombre que se corresponde con las tres etapas en las que se desarrollan. La primera etapa de estos métodos, la *interestructura*, tiene como objetivo el estudio de la relación entre las tablas que forman la tabla global, a través de los operadores que las representan. La segunda, el *compromiso*, busca obtener una nube media (ponderada) que represente la estructura común de esas tablas. El análisis de las relaciones entre individuos y variables en la tabla-compromiso es el objetivo de la tercera etapa (*intraestructura*).

Los métodos asimétricos más conocidos son el *Statis* (L'Hermeir des plantes, 1976) y el *Análisis Factorial Múltiple* (Escofier and Pages, 1984).

Los métodos simétricos se identifican con los llamados *Métodos de componentes* y se caracterizan por ajustar modelos que reproduzcan, de la manera más fiable posible, el dato original. Entre estos podemos citar los métodos de Tuckals (Kroonen-

berg and de Leeuw, 1980), basados en el modelo de Tucker (1966) y los modelos $CANDECOMP/PARAFAC$ (Carroll and Chang, 1970; Harshman, 1970) entre otros.

En este capítulo nos vamos a centrar en el estudio de los métodos simétricos.

2.2. NOTACIÓN

Toda la notación que se utiliza en este trabajo está tomada de Kiers (2000).

Para representar una matriz de datos en estadística normalmente se utiliza la letra X. Si la matriz es de dos vías la letra aparece en negrita $\mathbf{X}$ y en el caso de matrices de tres vías la notación utilizada es $\underline{\mathbf{X}}$.

Los elementos de una matriz de tres vías $\underline{\mathbf{X}}$ se denotan por x_{ijk} donde los índices toman valores entre 1 y su correspondiente letra en mayúscula: $i = 1,...,I$, $j = 1,...,J$, $k = 1,...,K$.

Las matrices de datos de n vías también se denominan **tensores**, por tanto un vector de orden I es un tensor en $\Re^I$, una matriz $I \times J$ es un tensor en $\Re^{I \times J}$ y una matriz de tres vías es un tensor en $\Re^{I \times J \times K}$, etc.

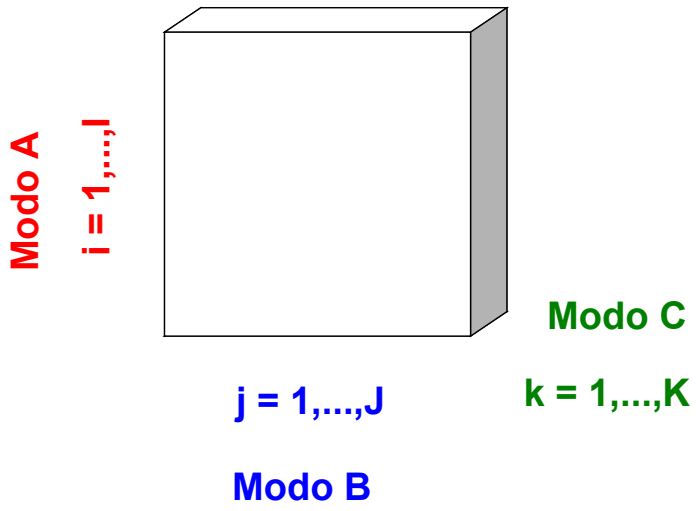


Figura 2.1: Modos de una matriz de tres vías.

Usaremos el termino **modo** para referirnos a cada una de las entidades de la matriz. Así en una matriz de tres vías el primer modo se denota como modo A (individuos), el segundo modo como modo B (variables) y el tercero como modo C (ocasiones) (Figura

2.1).

En la Figura 2.2 se muestra una matriz de tres vías y sus posibles cortes horizontales, verticales y frontales.

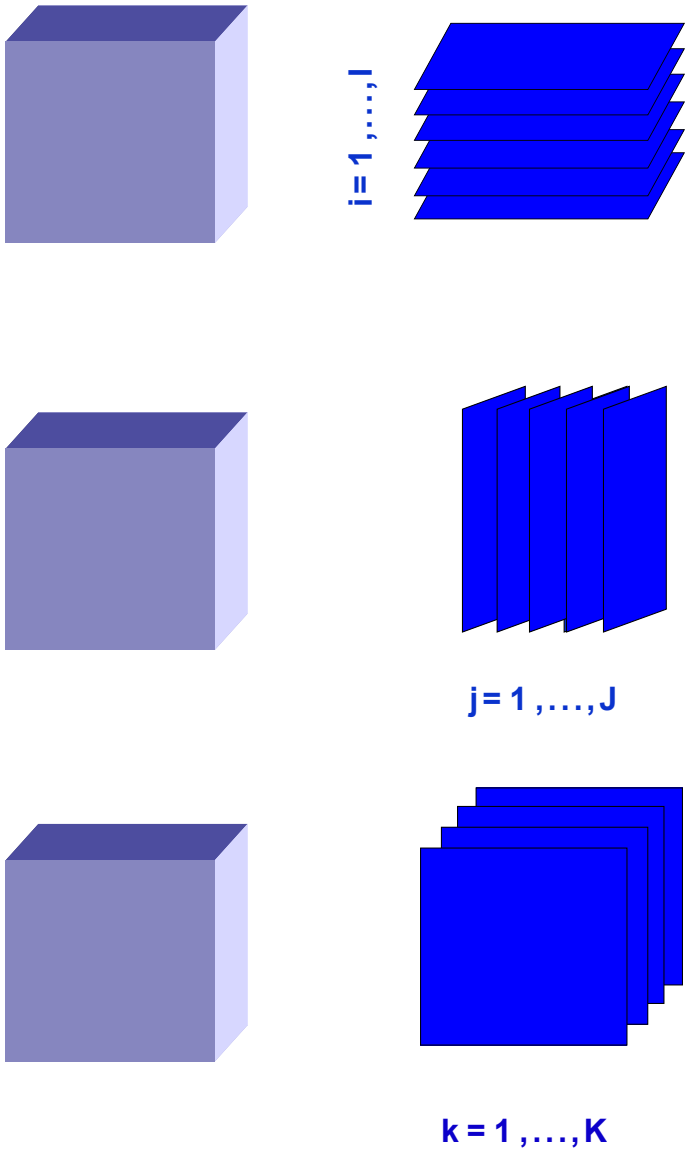


Figura 2.2: Cortes de una matriz de tres vías.

Un tensor $X \in \Re_{I_1 \times I_2 \times \ldots \times I_N}$ es **diagonal** si $x_{i_1 i_2 \ldots i_N} \neq 0$ solo si $i_1 = i_2 = \ldots = i_N$.

La transformación de la matriz de tres vías en una matriz de dos vías se conoce con el término de **matriciación** (en ocasiones también se utiliza el término **unfolding**). La matriciación del modo A: X_a conserva el primer modo (individuos) en filas, mientras que el segundo y el tercer modo (variables y condiciones) se combinan en columnas. Las matriciaciones de los modos B y C se pueden desarrollar de forma equivalente conservando los modos B y C respectivamente en filas y confundiendo los modos restantes en columnas, según sea el caso (Figura 2.3).

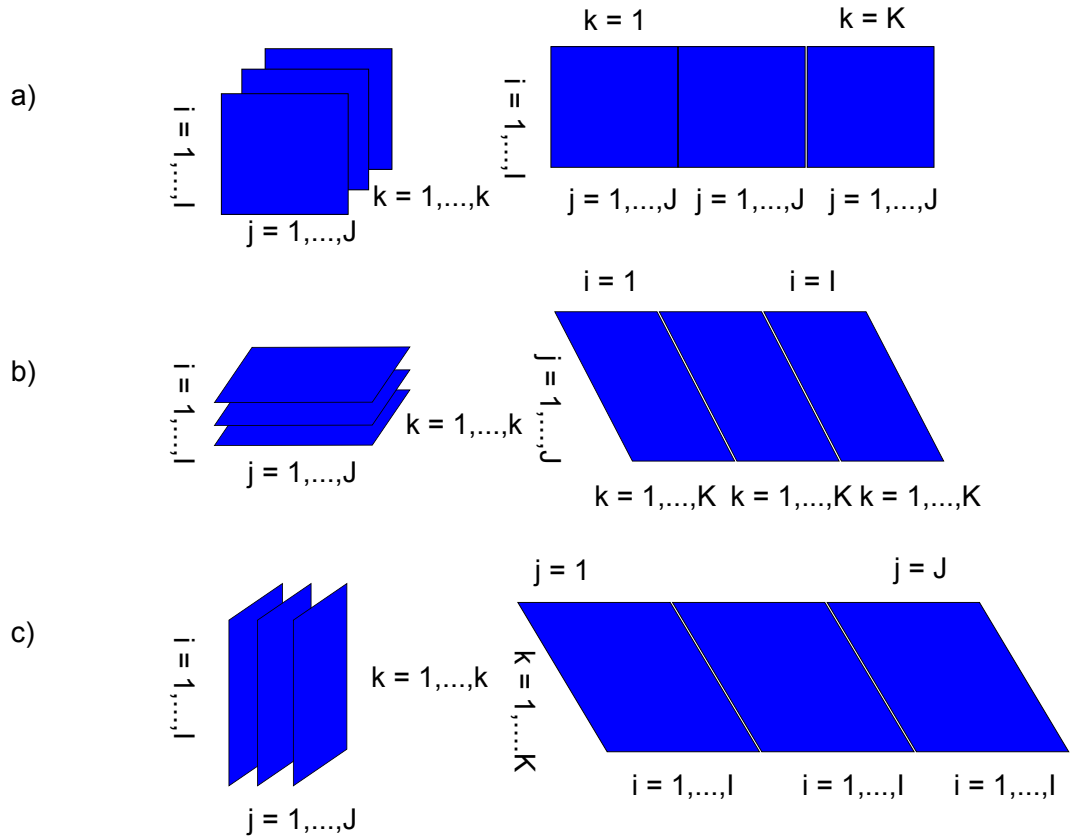


Figura 2.3: Matriciación de una matriz de tres vías: (a) modo A; (b) modo B; (c) modo C.

En algunas ocasiones también es conveniente representar todos los elementos de la

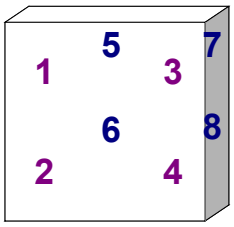


Figura 2.4: Ejemplo de matriciación.

matriz de tres vías como un vector, es decir **vectorizar**. Para vectorizar una matriz de tres vías o de orden superior simplemente se vectoriza el modo A y se obtiene $x = Vec(X_a)$.

La matriciación de la Figura 2.4 sería la siguiente:

$$X_a = \begin{pmatrix} 1 & 3 & 5 & 7 \\ 2 & 4 & 6 & 8 \end{pmatrix} \; X_b = \begin{pmatrix} 1 & 2 & 5 & 6 \\ 3 & 4 & 7 & 8 \end{pmatrix}$$

$$X_c = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \end{pmatrix} \; Vec(X) = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \end{pmatrix}$$

En el análisis de datos de múltiples vías se utilizan habitualmente una serie de operadores que se resumen a continuación:

- El **producto de Kronecker** se denota por el símbolo $\otimes$ y se define de la siguiente forma: $(\mathbf{U} \otimes \mathbf{V})_{ik,jl} = u_{ij}v_{kl}$. Por tanto tenemos:

$$\mathbf{U} \otimes \mathbf{V} = \begin{pmatrix} u_{11}\mathbf{V} & \ldots & u_{1J}\mathbf{V} \\ \ldots & \ldots & \ldots \\ u_{I1}\mathbf{V} & \ldots & u_{IJ}\mathbf{V} \end{pmatrix} \tag{2.1}$$

Ejemplo de producto de Kronecker:

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \otimes \begin{bmatrix} 0 & 5 \\ 6 & 7 \end{bmatrix} = \begin{bmatrix} 1\cdot 0 & 1\cdot 5 & 2\cdot 0 & 2\cdot 5 \\ 1\cdot 6 & 1\cdot 7 & 2\cdot 6 & 2\cdot 7 \\ 3\cdot 0 & 3\cdot 5 & 4\cdot 0 & 4\cdot 5 \\ 3\cdot 6 & 3\cdot 7 & 4\cdot 6 & 4\cdot 7 \end{bmatrix} = \begin{bmatrix} 0 & 5 & 0 & 10 \\ 6 & 7 & 12 & 14 \\ 0 & 15 & 0 & 20 \\ 18 & 21 & 24 & 28 \end{bmatrix}$$

- El **producto de Kronecker por columnas o producto de Khatri-Rao** (Rao and Mitra, 1971; McDonald, 1980) se denota por el símbolo $\bigodot$ y sólo es válido cuando las matrices tienen el mismo número de columnas L. Se define como: $(\mathbf{U} \bigodot \mathbf{V})_{ik,l} = u_{il}v_{kl}$. Por tanto tenemos:

$$\mathbf{U} \odot \mathbf{V} = (u_1 \otimes v_1 ... u_L \otimes v_L) \tag{2.2}$$

Ejemplo de producto de Khatri-Rao:

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \odot \begin{bmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 9 \end{bmatrix} = \begin{bmatrix} 1 & 8 & 21 \\ 2 & 10 & 24 \\ 3 & 12 & 27 \\ 4 & 20 & 42 \\ 8 & 25 & 48 \\ 12 & 30 & 54 \\ 7 & 32 & 63 \\ 14 & 40 & 72 \\ 21 & 48 & 81 \end{bmatrix}$$

- El **producto de Hadamard**: Dadas dos matrices **A** y **B**, ambas de tamaño $I \times J$, su producto Hadamard es también de tamaño $I \times J$ y se denota por $A * B$. Se

define como:

$$\mathbf{A} * \mathbf{B} = \begin{pmatrix} a_{11}b_{11} & ... & a_{1J}v_{1J} \\ ... & ... & ... \\ a_{I1}b_{I1} & ... & a_{IJ}b_{IJ} \end{pmatrix} \tag{2.3}$$

2.3. MODELOS

2.3.1. *CANDECOMP/PARAFAC*

Existen diferentes nombres para referirse a la descomposición *CANDECOMP/PARAFAC*. En la siguiente Tabla se recogen estos nombres y quien los propuso:

Nombre	Propuesto por
Polyadic form of a tensor	Hitchcock (1927)
PARAFAC (parallel factors)	Harshman (1970)
CANDECOMP o CAND (canonical decomposition)	Carroll and Chang (1970)
Topographic components model	Mocks (1988)
CP (CANDECOMP/PARAFAC)	Kiers (2000)

La descomposición CP factoriza un tensor en la suma de R tensores de rango uno. Por ejemplo, dado un tensor $\underline{\mathbf{X}} \in \Re^{I \times J \times K}$ la descomposición CP sería de la forma:

$$\underline{\mathbf{X}} \simeq \sum_{r=1}^{R} a_r \circ b_r \circ c_r \tag{2.4}$$

con $a_r \in \Re^I$, $b_r \in \Re^J$ y $c_r \in \Re^K$ para $r = 1, ..., R$. Equivalentemente:

$$x_{ijk} \simeq \sum_{r=1}^{R} a_{ir} b_{jr} c_{kr} \tag{2.5}$$

con $i = 1, ..., I$, $j = 1, ..., J$, $k = 1, ..., K$.

La Figura 2.5 ilustra lo explicado anteriormente.

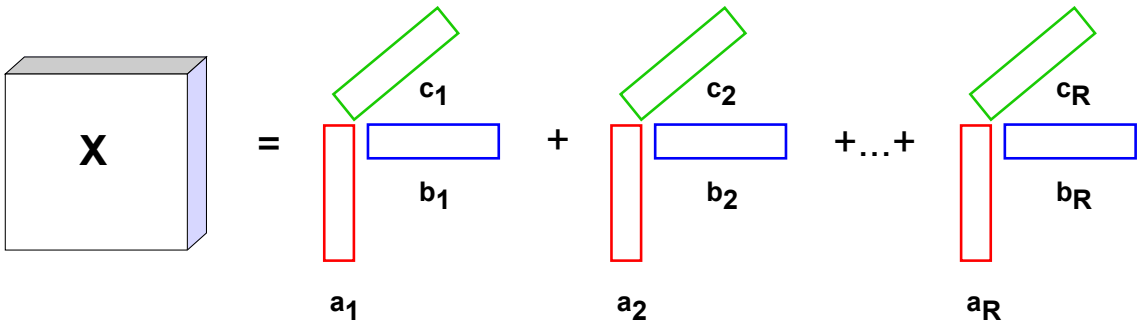


Figura 2.5: Descomposición CP de una matriz de tres vías.

También podemos escribirlo en notación matricial como:

$$\mathbf{X}_a \simeq \mathbf{A}(\mathbf{C} \odot \mathbf{B})' \tag{2.6}$$

$$\mathbf{X}_b \simeq \mathbf{B}(\mathbf{A} \odot \mathbf{C})' \tag{2.7}$$

$$\mathbf{X}_c \simeq \mathbf{C}(\mathbf{B} \odot \mathbf{A})' \tag{2.8}$$

donde $\mathbf{A}$ es de tamaño $(I \times R)$, $\mathbf{B}$ es de tamaño $(J \times R)$, $\mathbf{C}$ es de tamaño $(K \times R)$ y $\odot$ es el producto de Khatri-Rao.

Siguiendo a Kolda (2006) el modelo CP se puede expresar como:

$$\underline{\mathbf{X}} \simeq [[\mathbf{A}, \mathbf{B}, \mathbf{C}]] \equiv \sum_{r=1}^{R} a_r \circ b_r \circ c_r \tag{2.9}$$

A menudo es útil considerar que las columnas de $\mathbf{A}$, $\mathbf{B}$ y $\mathbf{C}$ están normalizadas y que los pesos los absorbe un vector $\lambda \in \Re^R$ de tal manera que:

$$\underline{\mathbf{X}} \simeq [[\lambda; \mathbf{A}, \mathbf{B}, \mathbf{C}]] \equiv \sum_{r=1}^{R} \lambda_r a_r \circ b_r \circ c_r \tag{2.10}$$

Para un tensor de orden N, $\mathbf{X} \in \Re^{I_1 \times I_2 \times ... \times I_N}$ la descomposición CP sería:

$$\mathbf{X} \simeq [[\lambda; \mathbf{A}^{(1)}, \mathbf{A}^{(2)}, ..., \mathbf{A}^{(N)}]] \equiv \sum_{r=1}^{R} \lambda_r a_r^{(1)} \circ a_r^{(2)} \circ ... \circ a_r^{(N)} \tag{2.11}$$

donde $\lambda \in \Re^R$ y $\mathbf{A}^{(n)} \in \Re^{I_n \times R}$ para n = 1,...,N. En este caso la forma matricial estaría dada por:

$$\mathbf{X}_n \simeq \mathbf{A}^{(n)} \wedge (\mathbf{A}^{(N)} \odot ... \odot \mathbf{A}^{(n+1)} \odot \mathbf{A}^{(n-1)} \odot ... \odot \mathbf{A}^{(1)})' \tag{2.12}$$

donde $\wedge = diag(\lambda)$.

Se define el **rango** de un tensor $\mathbf{X}$ como el menor número de tensores de rango uno tal que su suma genera $\mathbf{X}$ (Hitchcock, 1927; Kruskal, 1977). En otras palabras, el menor número de componentes en una descomposición CP exacta, donde exacta significa la igualdad en la ecuación 2.4.

La definición de rango de un tensor es análoga a la definición de rango de una matriz, pero sus propiedades son distintas. Una diferencia importante es que no existe un algoritmo sencillo que determine el rango de un tensor. En la practica, el rango de un tensor se determina numericamente ajustando varios modelos CP de rango R.

En general se verifica que si $\mathbf{X} \in \Re^{I \times J \times K}$ es un tensor entonces (Kruskal, 1989):

$$rango(\underline{\mathbf{X}}) \leq min(IJ, IK, JK) \tag{2.13}$$

La descomposición CP es única bajo ciertas condiciones. Sea $\mathbf{X} \in \Re^{I \times J \times K}$ un tensor de orden tres de rango R, es decir

$$\underline{\mathbf{X}} = [[\mathbf{A}, \mathbf{B}, \mathbf{C}]] = \sum_{r=1}^{R} a_r \circ b_r \circ c_r \tag{2.14}$$

Por **unicidad** entendemos que esta es la única combinación posible de tensores de rango uno cuya suma es igual a $\mathbf{X}$, con la excepción de las indeterminaciones elementales por escala y permutación.

El resultado más general y conocido sobre unicidad del modelo CP se debe a Kruskal (Kruskal, 1977, 1989) y se basa en el concepto de rango de una matriz. Kruskal dice que una condición suficiente para que la descomposición CP sea única es que:

$$k_a + k_b + k_c > 2R + 2 \tag{2.15}$$

siendo k_a, k_b y k_c los rangos de las matrices A, B y C respectivamente.

Sidiropoulos and Bro (2000) extienden el resultado de Kruskal a tensores de N vías. Sea $\mathbf{X}$ un tensor de N vías de rango R, supongamos que su descomposición CP es:

$$\mathbf{X} = \sum_{r=1}^{R} a_r^{(1)} \circ a_r^{(2)} \circ ... \circ a_r^{(N)} = [[\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, ..., \mathbf{A}^{(N)}]] \tag{2.16}$$

Entonces una condición suficiente para la unicidad es:

$$\sum_{n=1}^{N} k_{A^{(n)}} \geq 2R + (N - 1) \tag{2.17}$$

Todos los resultados anteriores proporcionan condiciones suficientes. Ten Berge and Sidiriopolous (2002) demuestran que la condición suficiente es también necesaria para tensores de rango R = 2 y R = 3, pero no para R > 3.

Como se ha visto anteriormente el modelo PARAFAC es restrictivo, por lo que los datos pueden no ajustarse al modelo. La falta de concordancia entre los datos y el modelo hará que el algoritmo tenga problemas para dar con una solución válida dando lugar a **soluciones degeneradas**. Los primeros estudios sobre soluciones degeneradas son de Harshman and Lundy (1984). Estudios posteriores se deben a Kruskal et al. (1989), Paatero (2000) y Stegeman (2006). Zijlstra and Kiers (2002) afirman que sólo los modelos con solución única pueden tener soluciones degeneradas. Harshman (2005) afirma que existen tres tipos de soluciones degeneradas. El primer tipo lo denomina *temporary degeneracy*, no es realmente degeneración, sólo lo parece porque el algoritmo está atascado en una ciénaga. Se puede comparar con el altiplano que se encuentra por casualidad el es-

calador que lleva los ojos vendados. En el segundo tipo denominado *bounded degeneracy* la solución es degenerada pero estable en el sentido de que comenzando desde distintos puntos de partida se llega a la misma solución degenerada. El tercer tipo denominado *divergent degeneracy* es el más problematico. La solución óptima se aleja porque no existe dicha solución, la idoneidad de cada solución puede mejorarse aumentando el número de iteraciones que en realidad hacen que la solución sea más degenerada. La causa de este tipo de degeneración es que los datos no se ajustan al modelo. Los datos tienen variación Tucker, es decir, la componente a_s que debería tener relación exclusivamente con la componente b_s tiene también relación con $b_s^{'}$. En principio, las soluciones degeneradas no se deben interpretar.

Krijnen and Kroonenberg (2000) discuten una serie de medidas para evaluar si un algoritmo tiende a una solución degenerada. Para evaluar si la solución es degenerada se necesitan los cosenos entre dos componentes de un modo. Por ejemplo el coseno entre la componente s y s' del primer modo es $cos(\alpha_{s,s'}) = (a_s^{'} a_{s'})$. Si definimos f_s como $a_s \otimes b_s \otimes c_s$, entonces el coseno $\theta_{ss'}$ entre f_s y $f_{s'}$ es igual a:

$$cos(\theta_{ss'}) = cos(\alpha_{ss'}) \times cos(\beta_{ss'}) \times cos(\gamma_{ss'}) \tag{2.18}$$

Si el triple coseno es aproximadamente -1 es casi seguro que se trate de una solución degenerada. En este caso las dos componentes f_s y $f_{s'}$ son proporcionales lo cual está terminantemente prohibido en un modelo PARAFAC. Esta conclusión también puede apoyarse mediante la creación de una matriz Θ de tamaño S $\times$ S de los $cos(\theta_{ss'})$ e inspeccionar el menor valor propio. Si este está por debajo de 0.50 podríamos estar ante una solución degenerada. Además se debe valorar el *condition number* (es decir, el mayor valor propio divido entre el más pequeño) de la matriz de triples cosenos Θ. Si el *condition number* es grande, digamos mayor que 5, es posible una degeneración. Tanto el valor propio más pequeño como el *condition number* son indicadores de si la matriz de cosenos triples es de rango completo como debe ocurrir en un modelo PARAFAC.

Si se sospecha que estamos en presencia de una solución degenerada deberíamos asegurarnos realizando el análisis al menos una vez más con muchas más iteracciones y comprobar si los indicadores anteriores son aún más extremos. Además se debería

probar con distintas soluciones iniciales. Para eliminar la degeneración se debe realizar el análisis con menos componentes, reexaminar el procedimiento de preprocesamiento que se ha seguido, o imponer restricciones sobre las componentes que pueden prevenir que exista degeneración, en particular, que uno de los modos sea ortogonal o no negativo. Si todo esto falla siempre es posible elegir un modelo *Tucker*.

2.3.2. MODELOS *TUCKER*

La descomposición *Tucker* fue introducida por Tucker (1966). Al igual que ocurre con la descomposición *CP* existen diferentes nombres para referirnos a la descomposición *Tucker* que se muestran en la siguiente Tabla:

Nombre	Propuesto por
Three-mode factor analysis (3MFA/Tucker3)	Tucker (1966)
Three-mode PCA (3MPCA)	Kroonenberg and de Leeuw (1980)
N-mode PCA	Kapteyn et al. (1986)
Higher-order SVD (HOSVD)	Lathauwer et al. (2000)
N-mode SVD	Vasilescu and Terzopoulos (2002)

La descomposición de *Tucker* es una forma de *PCA* de orden superior. Descompone un tensor en un core tensor multiplicado por una matriz a lo largo de cada modo. Por tanto, en una matriz de tres vías donde $\underline{\mathbf{X}} \in \Re^{I \times J \times K}$ tenemos:

$$\underline{\mathbf{X}} \simeq \sum_{p=1}^{P} \sum_{q=1}^{Q} \sum_{r=1}^{R} g_{pqr} a_p \circ b_q \circ c_r = [[\underline{\mathbf{G}}; \mathbf{A}, \mathbf{B}, \mathbf{C}]] \tag{2.19}$$

Donde $\mathbf{A} \in \Re^{I \times P}$, $\mathbf{B} \in \Re^{J \times Q}$ y $\mathbf{C} \in \Re^{K \times R}$ pueden entenderse como las componentes principales de cada modo. El tensor $\underline{\mathbf{G}} \in \Re^{P \times Q \times R}$ se llama core tensor y muestra el nivel de interacción entre las diferentes componentes. La última igualdad usa la notación introducida por Kolda (2006).

Equivalentemente la descomposición de *Tucker* se puede escribir como:

$$x_{ijk} \simeq \sum_{p=1}^{P} \sum_{q=1}^{Q} \sum_{r=1}^{R} g_{pqr} a_{ip} b_{jq} c_{kr} \tag{2.20}$$

para $i = 1, ..., I$, $j = 1, ..., J$, $k = 1, ..., K$. P, Q y R son el número de componentes (es decir, columnas) de las matrices $\mathbf{A}$, $\mathbf{B}$ y $\mathbf{C}$ respectivamente. La descomposición de *Tucker* se muestra en la Figura 2.6.

.

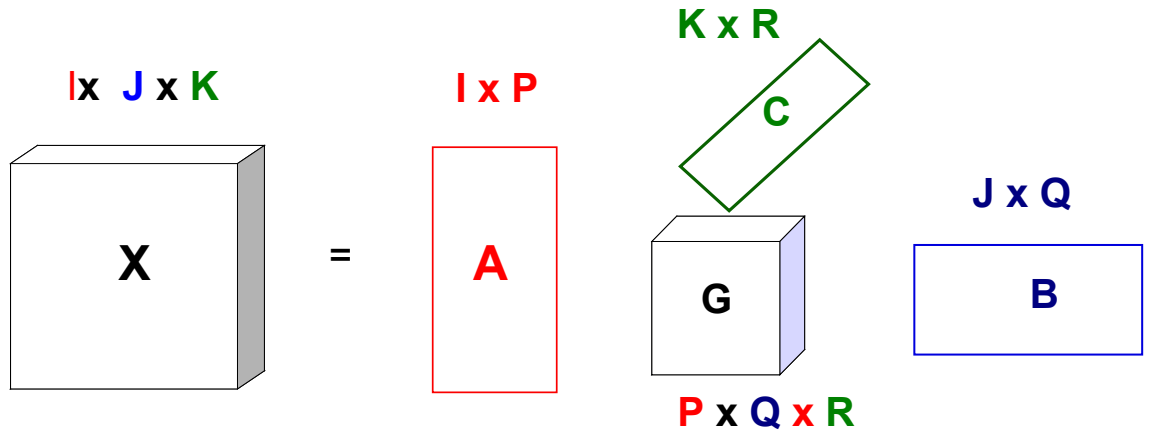


Figura 2.6: Descomposición *Tucker* de una matriz de tres vías.

La mayoría de los algorítmos asumen que las matrices **A**, **B** y **C** son ortonormales, pero este requisito no es obligatorio. La descomposición CP se puede ver como un caso especial de *Tucker* donde el core tensor is superdiagonal y P=Q=R. La forma matricial del modelo es:

$$\mathbf{X}_a \simeq \mathbf{A}\mathbf{G}_a(\mathbf{C} \otimes \mathbf{B})' \tag{2.21}$$

$$\mathbf{X}_b \simeq \mathbf{B}\mathbf{G}_b(\mathbf{A} \otimes \mathbf{C})' \tag{2.22}$$

$$\mathbf{X}_c \simeq \mathbf{C}\mathbf{G}_c(\mathbf{B} \otimes \mathbf{A})' \tag{2.23}$$

Existen dos variantes de la descomposición de *Tucker*. En la descomposición *Tucker*2 (Tucker, 1966) de una matriz de tres vías una de las matrices (A, B ó C) es igual a la matriz identidad. Por tanto el modelo *Tucker*2 es de la forma:

$$\underline{\mathbf{X}} = [[\underline{\mathbf{G}}; \mathbf{A}, \mathbf{B}, \mathbf{I}]] \tag{2.24}$$

Esta ecuación es la misma que la ecuación 2.19 excepto que $\underline{\mathbf{G}} \in \Re^{P \times Q \times R}$ con R = K y $\mathbf{C} = \mathbf{I}$, la matriz identidad $K \times K$.

Por otro lado en la descomposición *Tucker*1 dos de las matrices son iguales a la matriz identidad. Por ejemplo, si la segunda y la tercera matriz son iguales a la matriz

identidad tenemos:

$$\underline{\mathbf{X}} = [[\underline{\mathbf{G}}; \mathbf{A}, \mathbf{I}, \mathbf{I}]] \tag{2.25}$$

Unicidad

La descomposición *Tucker* no es única. Sea el modelo *Tucker*3 (P, Q, R) de la matriz de tres vías $\underline{\mathbf{X}}$ $(I \times J \times K)$:

$$\mathbf{X} = \mathbf{AG}(\mathbf{C}^{'} \otimes \mathbf{B}^{'}) + \mathbf{E}$$

donde $\mathbf{X}$ $(I \times JK)$, $\mathbf{G}$ $(P \times QR)$ y $\mathbf{E}$ $(I \times JK)$ son las matriciaciones de $\underline{\mathbf{X}}$, $\underline{\mathbf{G}}$ y $\underline{\mathbf{E}}$ respectivamente y $\mathbf{A}$ $(I \times P)$, $\mathbf{B}$ $(J \times Q)$, $\mathbf{C}$ $(K \times R)$ son las matrices de componentes. Si $\mathbf{S}$ $(P \times P)$ es una matriz no singular, entonces se verifica:

$$\mathbf{X} = \mathbf{ASS}^{-1}\mathbf{G}(\mathbf{C}^{'} \otimes \mathbf{B}^{'}) + \mathbf{E} = \tilde{\mathbf{A}}\tilde{\mathbf{G}}(\mathbf{C}^{'} \otimes \mathbf{B}^{'}) + \mathbf{E} \tag{2.26}$$

Por lo tanto la transformación con $\mathbf{S}$ no modifica el ajuste y se tienen además los mismos residuales. El modelo con parámetros $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ y $\mathbf{G}$ es equivalente al modelo con parámetros $\tilde{\mathbf{A}}$, $\mathbf{B}$, $\mathbf{C}$ y $\tilde{\mathbf{G}}$ y por tanto la descomposición *Tucker* no es única.

Cuando la matriz $\mathbf{S}$ es ortogonal $(\mathbf{S}^{'}\mathbf{S} = \mathbf{SS}^{'} = \mathbf{I})$, entonces la transformación se denomina transformación ortogonal. La propiedad descrita en 2.26 se conoce como libertad rotacional.

Podemos aprovechar la libertad rotacional para conseguir un modelo con matrices de componentes $\mathbf{A}$, $\mathbf{B}$ y $\mathbf{C}$ ortonormales por columnas. Esto supone una ventaja a la hora de interpretar los elementos de la core matrix, dado que entonces el valor del elemento g_{pqr} elevado al cuadrado es igual a la cantidad de variabilidad de $\underline{\mathbf{X}}$ explicada por la combinación de componentes a_p, b_q y c_r (Kroonenberg, 1983).

Esto es así debido a la propiedad de las matrices ortogonales por columnas:

$$\| \mathbf{X} \| = \| \mathbf{AX} \| \tag{2.27}$$

donde $\mathbf{X}$ es una matriz $(I \times J)$ cualquiera y $\mathbf{A}$ es una matriz ortogonal por columnas $(K \times I)$. Por lo tanto, escribiendo el modelo $Tucker3$ con matrices de componentes ortogonales, se verifica que:

$$\| \hat{\mathbf{X}} \| = \| \mathbf{AG}(\mathbf{C}' \otimes \mathbf{B}') \| = \| \mathbf{G} \| \tag{2.28}$$

Rotaciones

Esta libertad rotacional abre la puerta a la posibilidad de elegir transformaciones que simplifiquen la estructura de la core matrix, de manera que la mayoría de los elementos de la misma sean cero, y por tanto las interacciones entre dichas componentes se eliminen.

Supongamos un modelo $Tucker3$ (4,4,4), la core matrix tiene 64 elementos y por tanto 60 interacciones, lo que dificulta enormemente su interpretación. Si todos los elementos de la core matrix son de tamaño significativo será necesario interpretar todas las combinaciones de componentes. De ahí la importancia de conseguir una rotación de la core matrix (con las correspondientes rotaciones para las matrices de componentes), de manera que sean significativos sólo unos pocos elementos de la core matrix. Existen tres cuestiones que se deben tener en cuenta:

- Tipo de rotación (oblicua u ortogonal).
- Tipo de criterio (elementos predefinidos cero, superdiagonalidad o simplemente simplificar la estructura).
- Grado de pérdida de ajuste (no hay pérdida de ajuste o si hay pérdida de ajuste).

Por tanto la primera elección que debe hacerse es si la rotación va a ser ortogonal u oblicua. Las rotaciones ortogonales tienen como principal ventaja que las matrices de componentes permanecen ortogonales después de la rotación, lo que facilita la interpretación del modelo. Además las transformaciones ortogonales evitan problemas con soluciones degeneradas y son más sencillas y fáciles de implementar (Kiers, 1992). Además rotar sucesivamente un modelo hacia la superdiagonalidad usando transformaciones ortogonales tiene muchas ventajas desde el punto de vista de la interpretación. Esto es

debido a que se consigue una visualización del modelo más directa ya que tenemos matrices de componentes ortogonales y elementos significativos en la superdiagonal de la core. Por todas estas razones se utilizan habitualmente rotaciones ortogonales aunque es necesario reconocer que las rotaciones ortogonales restringen el espacio de las soluciones en comparación con las rotaciones obliquas.

Otra posibilidad es intentar transformar la core matrix en una estructura donde sólo los elementos de la superdiagonal sean distintos de cero (suponiendo P = Q = R). También es posible intentar obtener frontal slices de la core matrix diagonales. Si todos los frontal slices de la core matrix se pueden diagonalizar simultameamente entonces el modelo $Tucker3$ (R,R,R) es equivalente al modelo $PARAFAC$ con R componentes. Finalmente es común intentar rotar la core matrix para conseguir que unos pocos elementos en valor absoluto sean grandes y el resto tenga valores pequeños.

El tercer aspecto que se debe tener en cuenta es si se permite o no pérdida en el ajuste. En algunas ocasiones es preferible sacrificar ajuste para introducir ceros en la core matrix.

Modelos Tucker para N-Vías

A pesar de que la descomposición $Tucker$ fue introducida en el contexto de las tres vías, el modelo $Tucker$ se puede generalizar a tensores de N-vías como:

$$\mathbf{X} = [[\mathbf{G}; \mathbf{A}^{(1)}, \mathbf{A}^{(2)}, ..., \mathbf{A}^{(N)}]] \tag{2.29}$$

o equivalentemente como:

$$x_{i_1 i_2 ... i_N} = \sum_{r_1=1}^{R_1} \sum_{r_2=1}^{R_2} ... \sum_{r_N=1}^{R_N} g_{r_1 r_2 ... r_N} a^{(1)}_{i_1 r_1} a^{(2)}_{i_2 r_2} ... a^{(N)}_{i_N r_N} \tag{2.30}$$

La forma matricial es:

$$\mathbf{X}_{(n)} = \mathbf{A}^{(n)} \mathbf{G}_{(n)} (\mathbf{A}^{(N)} \otimes ... \otimes \mathbf{A}^{(n+1)} \otimes \mathbf{A}^{(n-1)} \otimes ... \otimes \mathbf{A}^{(1)})' \tag{2.31}$$

2.3.3. OTRAS DESCOMPOSICIONES

Existen otras descomposiciones relacionadas con las decomposiciones CP y $Tucker$. La mayoría de ellas nacen en el ambito de la psicometría y actualmente están siendo aplicadas a otros campos como la quimiometría. En la siguiente Tabla se muestran estas descomposiciones y por quien fueron propuestas.

Nombre	Propuesto por
Individual differences in scaling ($INDSCAL$)	Carroll and Chang (1970)
Parallel factors for cross products ($PARAFAC2$)	Harshman (1972)
CANDECOMP with linear constraints ($CANDELINC$)	Carroll et al. (1980)
Decomposition into directional components ($DEDICOM$)	Harshman (1978)
$PARAFAC$ and $Tucker2$ ($PARATUCK2$)	Harshman and Lundy (1996)

INDSCAL

El escalamiento de diferencias individuales ($INDSCAL$) es un caso especial de CP para tensores de tres vías que son simétricos en dos modos. Fue propuesto por Carroll and Chang (1970) en el mismo paper donde presentó la descomposición CANDECOMP.

En la descomposición $INDSCAL$ las dos primeras matrices de la descomposición deben ser iguales. Sin perdida de generalidad, supongamos que los dos primeros modos son simétricos. Por tanto para un tensor $X \in \Re^{I \times I \times K}$ con $x_{ijk} = x_{jik}$ para todos i,j,k, el modelo $INDSCAL$ está dado por:

$$\underline{\mathbf{X}} \simeq [[\mathbf{A}, \mathbf{A}, \mathbf{C}]] = \sum_{r=1}^{R} a_r \circ a_r \circ c_r \tag{2.32}$$

Las aplicaciones que involucran cortes simétricos son bastante comunes especialmente cuando se trata de medir similitudes, disimilitudes, distancias o matrices de covarianzas.

PARAFAC2

El *PARAFAC2* (Harshman, 1972) no es extrictamente una descomposición de tensores sino una variante de la descomposición *CP* que puede ser aplicada a una colección de matrices que tienen el mismo número de columnas y diferente número de filas. Aquí vamos a aplicar *PARAFAC2* a un conjunto de matrices X_k con k = 1,...,K donde cada X_k es de tamaño $I_k \times J$, donde I_k puede ser diferente para cada k.

La descomposición *PARAFAC2* relaja algunas de las restricciones de la descomposición *CP*. Así mientras que la descomposición *CP* utiliza los mismos factores para una colección de matrices, en la descomposición *PARAFAC2* se aplica el mismo factor para uno de los modos y permite que los otros factores puedan variar. Una ventaja del modelo *PARAFAC2* es que no solo permite aproximar los datos en tensores de tres vías con menos restricciones que la descomposición *CP* sino que además se puede aplicar a una colección de matrices de diferente tamaño para uno de los modos, por ejemplo, igual número de columnas pero diferente número de filas.

Sea R el número de dimensiones de la descomposición. Entonces el modelo *PARAFAC2* tiene la forma:

$$\mathbf{X}_k \simeq \mathbf{U}_k \mathbf{S}_k \mathbf{V}^{'} \tag{2.33}$$

para k = 1,...,K. Donde $\mathbf{U}_k$ es una matriz $I_k \times R$, $\mathbf{S}_k$ es una matriz diagonal $R \times R$ para k = 1,...,K y $\mathbf{V}$ es una matriz $J \times R$ que no varía con k. El modelo *PARAFAC2* para un conjunto de matrices con distintos tamaños se muestra en la Figura 2.7.

La descomposición *PARAFAC2* no es única sin restricciones adicionales. Por ejemplo, si $\mathbf{T}$ es una matriz no singular $R \times R$ y $\mathbf{F}_k$ es una matriz diagonal $R \times R$ para k = 1,...,K., entonces

$$\mathbf{U}_k \mathbf{S}_k \mathbf{V}' = (\mathbf{U}_k \mathbf{S}_k \mathbf{T}^{-1} \mathbf{F}_k^{-1}) \mathbf{F}_k (\mathbf{V}\mathbf{T}^{'})' = \mathbf{G}_k \mathbf{F}_k \mathbf{W}^{'}$$

es tambien una descomposición válida. Para conseguir la unicidad Harshman (1972) impone la restricción siguiente: $\mathbf{U}_k^{'}\mathbf{U}_k$ ha de ser constante para todo k, es decir $\phi = \mathbf{U}_k^{'}\mathbf{U}_k$

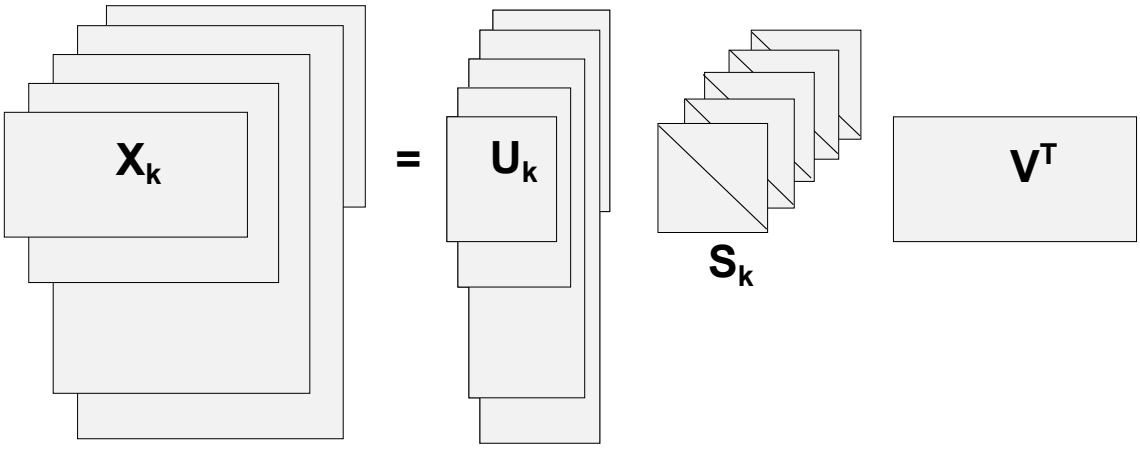


Figura 2.7: Modelo $PARAFAC2$.

para k = 1,...,K. Por tanto con esta restricción el modelo $PARAFAC2$ se puede escribir como:

$$\mathbf{X}_k \simeq \mathbf{Q}_k \mathbf{H} \mathbf{S}_k \mathbf{V}^{'} \tag{2.34}$$

para k = 1,...,K. Aquí, $\mathbf{U}_k = \mathbf{Q}_k \mathbf{H}$, donde $\mathbf{Q}_k$ es una matriz de tamaño $I_k \times R$ ortonormal y $\mathbf{H}$ es una matriz $R \times R$.

$CANDELINC$

Un problema importante en el análisis multidimensional es la interpretación de las matrices factor resultantes de la descomposición. Esta interpretación se puede facilitar mediante la introducción de restricciones. $CANDELINC$ (canonical decomposition with linear constraints) es una descomposición CP con restricciones lineales en una o más matrices que fué introducida por Carroll, Pruzansky, and Kruskal (1980).

En el caso de las matrices de tres vías, $CANDELINC$ requiere que las matrices factor de la descomposición CP verifiquen:
$\mathbf{A} = \Phi_A \hat{A}$, $\mathbf{B} = \Phi_B \hat{B}$, $\mathbf{C} = \Phi_C \hat{C}$. Donde $\Phi_A \in \Re^{I \times M}, \Phi_B \in \Re^{J \times N}$ y $\Phi_C \in \Re^{K \times P}$ define el espacio de las columnas para cada matriz y $\hat{A}, \hat{B}$ y $\hat{C}$ son las matrices de restricciones.

Por tanto el modelo $CANDELINC$ se puede expresar como:

$$\underline{\mathbf{X}} \simeq [[\Phi_A \hat{A}, \Phi_B \hat{B}, \Phi_C \hat{C}]] \tag{2.35}$$

Sin perdida de generalidad, podemos asumir que las matrices de restricciones Φ_A, Φ_B, Φ_C son ortonormales.

$DEDICOM$

$DEDICOM$ (decomposition into directional components) es una familia de descomposiciones introducida por Harshman (1978). Supongamos que tenemos I objetos y una matriz $\mathbf{X} \in \Re^{I \times I}$ que describe las relaciones asimétricas que existen entre ellos. Por ejemplo, los objetos podrían ser paises y x_{ij} el valor de las exportaciones del país i al país j. $DEDICOM$ agrupa los I objetos en R dimensiones latentes y describe sus patrones de interacción mediante el cálculo de $\mathbf{A} \in \Re^{I \times R}$ y $\mathbf{R} \in \Re^{R \times R}$ de tal forma que:

$$X \simeq ARA^{'} \tag{2.36}$$

Cada columna de $\mathbf{A}$ corresponde a una dimensión latente de manera que a_{ir} se corresponde con la participación del objeto i en el grupo r y la matriz $\mathbf{R}$ indica la interacción entre las diferentes componentes.

El $DEDICOM$ para tres vías es una extensión del modelo $DEDICOM$ que incorpora un tercer modo en los datos. Supongamos que $\underline{\mathbf{X}} \in \Re^{I \times I \times K}$, en el ejemplo anterior el tercer modo podrían ser los años. La descomposición en este caso sería:

$$\mathbf{X}_k = \mathbf{A}\mathbf{D}_k\mathbf{R}\mathbf{D}_k\mathbf{A}^{'} \tag{2.37}$$

para k = 1,...,K y $\mathbf{A}$ y $\mathbf{R}$ son del tipo de la ecuación 2.36. Las matrices $\mathbf{D}_k \in \Re^{R \times R}$ son diagonales y el valor $(\mathbf{D}_k)_{rr}$ indica la participación de la componente r en el tiempo k. La descomposición $DEDICOM$ para tres vías se muestra en la Figura 2.8

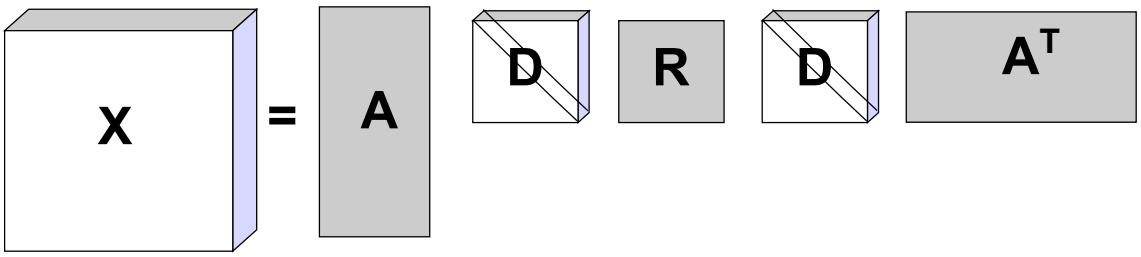


Figura 2.8: Modelo $DEDICOM$ para tres vías.

$PARATUCK2$

Harshman and Lundy (1996) proponen el modelo $PARATUCK2$ como una generalización del modelo $DEDICOM$ que considera interacciones entre dos conjuntos diferentes de objetos. El nombre es debido a que esta descomposición se puede considerar como una combinación de las descomposiciones CP y $Tucker2$.

Dado un tensor de orden tres $\underline{\mathbf{X}} \in \Re^{I \times J \times K}$, el objetivo es agrupar los objetos del primer modo en P componentes latentes y los objetos del modo dos en Q componentes latentes. Por tanto la descomposición $PARATUCK2$ está dada por:

$$\mathbf{X}_k \simeq \mathbf{A}\mathbf{D}_k^A\mathbf{R}\mathbf{D}_k^B\mathbf{B}' \tag{2.38}$$

para k = 1,...,K. Aquí, $\mathbf{A} \in \Re^{I \times P}$, $\mathbf{B} \in \Re^{J \times Q}$, $\mathbf{R} \in \Re^{P \times Q}$ y $\mathbf{D}_k^A \in \Re^{P \times P}$ y $\mathbf{D}_k^B \in \Re^{Q \times Q}$ son matrices diagonales. Como ocurre en el modelo $DEDICOM$, las columnas de $\mathbf{A}$ y de $\mathbf{B}$ corresponden a los factores latentes, por tanto b_{jq} se corresponde con la asociación del objeto j con la componente latente q. Por otro lado los elementos de las matrices diagonales indican el grado de participación de cada componente latente con respecto a la tercera dimensión. Finalmente, la matriz rectangular $\mathbf{R}$ representa la interacción entre la componente P en $\mathbf{A}$ y la componente latente Q en $\mathbf{B}$. El modelo $PARATUCK2$ se muestra en la Figura 2.9.

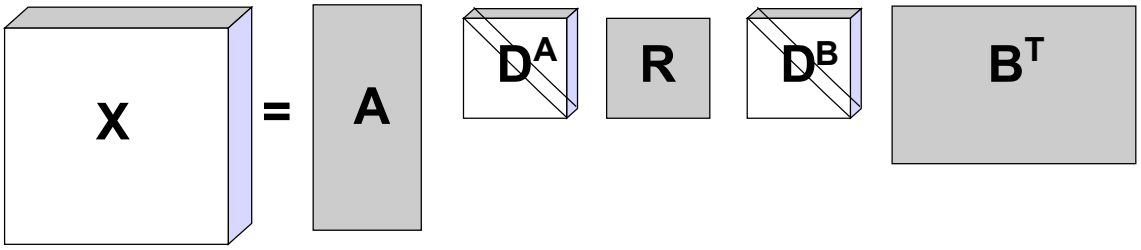
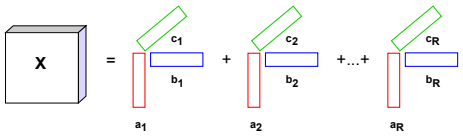
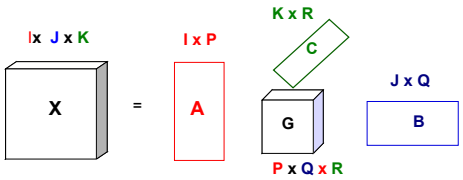
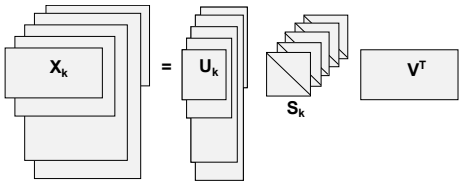
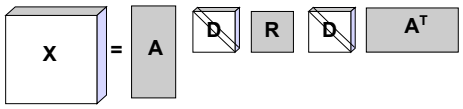
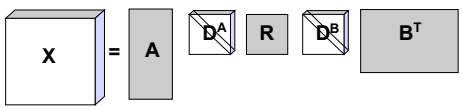


Figura 2.9: Modelo $PARATUCK2$.

2.3.4. TABLA RESUMEN

Modelo	Descomposición
PARAFAC/CANDECOMP (Carroll and Chang, 1970; Harshman, 1970)	$X \simeq \sum_{r=1}^{R} a_r \circ b_r \circ c_r$ En notación matricial: $X_a \simeq A(C \odot B)'$ $X_b \simeq B(A \odot C)'$ $X_c \simeq C(B \odot A)'$ 
Tucker (Tucker, 1966)	$X \simeq \sum_{p=1}^{P} \sum_{q=1}^{Q} \sum_{r=1}^{R} g_{pqr} a_p \circ b_q \circ c_r$ En notación matricial: $X_a \simeq AG_a(C \otimes B)'$ $X_b \simeq BG_b(A \otimes C)'$ $X_c \simeq CG_c(B \otimes A)'$ 
INDSCAL (Carroll and Chang, 1970)	Es un caso especial de *CP* para tensores de tres vías que son simétricos en dos modos: $X \simeq \sum_{r=1}^{R} a_r \circ a_r \circ c_r$

Modelo	**Descomposición**
*PARAFAC*2 (Harshman, 1972)	Es una variante de la descomposición *CP* que puede ser aplicada a una colección de matrices X_k que tienen el mismo número de columnas y diferente número de filas. $X_k \simeq U_k S_k V'$ 
CANDELINC (Carroll et al., 1980)	Es una descomposición *CP* con restricciones lineales en una o más matrices.
DEDICOM (Harshman, 1978)	Es una descomposición para matrices de tres vías de la forma $X \in \Re^{I \times I \times K}$. La descomposición en este caso es: $X \simeq A D_k R D_k A'$ 
*PARATUCK*2 (Harshman and Lundy, 1996)	Es una generalización del modelo *DEDICOM* para matrices $X \in \Re^{I \times J \times K}$. La descomposición en este caso es: $X \simeq A D_k^A R D_k^B B'$ 

2.4. PREPROCESAMIENTO DE LOS DATOS

Al igual que ocurre en el análisis de datos de dos vías es habitual preprocesar los datos antes de comenzar con el análisis. En el caso del análisis de datos de dos vías, los datos se suelen centrar y/o estandarizar para eliminar diferencias de nivel y/o de escala. En el caso de tres vías será necesario especificar a través/dentro de qué modo o modos se va a realizar el centrado y/o escalado de los datos (Bro, 1988).

El tipo de centrado se indica especificando el modo o modos a través de los cuales los datos se centran. Por ejemplo el centrado a través del modo A se efectúa promediando los datos para las entidades del modo A y restando la media obtenida a todos los datos:

$$\widetilde{x}_{ijk} = x_{ijk} - \frac{\sum_{i=1}^{I} x_{ijk}}{I} = x_{ijk} - \overline{x}_{\cdot jk} \tag{2.39}$$

Se puede realizar el centrado a través de cualquiera de los modos. Si se realiza el centrado a través de más de un modo se debe realizar primero el centrado a través de un modo y el resultado de este centrarlo a través del segundo modo. Este tipo de centrado se denomina doble centrado. El triple centrado consiste en centrar a través de los tres modos.

El escalado divide todos los elementos entre un término constante permitiendo eliminar diferencias de escala entre las variables. Para indicar el tipo de normalización utilizada se especifica el modo dentro del cual se normalizan los datos. Así escalar dentro de los elementos del modo B lleva a calcular el factor:

$$\upsilon_j = \sqrt{\sum_{i=1}^{I}\sum_{k=1}^{K} x_{ijk}^2 / IK} \tag{2.40}$$

y los datos escalados se calculan como:

$$\widetilde{x}_{ijk} = \frac{x_{ijk}}{\upsilon_j} \tag{2.41}$$

2.5. ALGORITMOS

Veamos ahora como ajustar los modelos que se han descrito anteriormente.

La mayoría de los algorítmos se basan en el algoritmo de los mínimos cuadrados alternados (ALS) (Yates, 1933). El algoritmo de los mínimos cuadrados alternados se basa en dividir los parámetros en grupos. Cada conjunto de parámetros se estima en el sentido de mínimos cuadrados condicionalmente sobre los parámetros restantes. La estimación de los parámetros se repite iterativamente hasta que los valores de los parámetros estimados no difiera significativamente (Bro, 1988).

2.5.1. MODELO *PARAFAC*

Suponiendo fijo el número de componentes, existen diversos algorítmos para realizar la descomposición CP. Aquí se va a desarrollar el algortmo de los mínimos cuadrados alternados (ALS) propuesto en los paper originales de Carroll and Chang (1970) y Harshman (1970).

Sea $\underline{\mathbf{X}} \in \Re^{I\times J\times K}$ un tensor de tres modos. El objetivo es calcular la descomposición CP de R componentes que mejor aproxime a $\underline{\mathbf{X}}$, es decir encontrar $\hat{\underline{\mathbf{X}}}$ con $\hat{\underline{\mathbf{X}}} = \sum_{r=1}^{R} \lambda_r a_r b_r c_r = [[\lambda; \mathbf{A}, \mathbf{B}, \mathbf{C}]]$ tal que:

$$\min_{\hat{X}} \|\underline{\mathbf{X}} - \hat{\underline{\mathbf{X}}}\| \tag{2.42}$$

El algoritmo de los mínimos cuadrados fija las matrices $\mathbf{B}$ y $\mathbf{C}$ para resolver $\mathbf{A}$, fija las matrices $\mathbf{A}$ y $\mathbf{C}$ para resolver $\mathbf{B}$ y fija las matrices $\mathbf{A}$ y $\mathbf{B}$ para resolver $\mathbf{C}$. Este procedimiento se repite hasta que se alcanza el criterio de convergencia. Los pasos a seguir son por tanto los siguientes:

Paso 0 Se inicializan $\mathbf{B}$ y $\mathbf{C}$

Paso 1 $\mathbf{A} = \mathbf{X}_a[(\mathbf{C} \odot \mathbf{B})^T]^\dagger$

Paso 2 $\mathbf{B} = \mathbf{X}_b[(\mathbf{C} \odot \mathbf{A})^T]^\dagger$

Paso 3 $\mathbf{C} = \mathbf{X}_c[(\mathbf{B} \odot \mathbf{A})^T]^\dagger$

Paso 4 Se comprueba el criterio de convergencia

Paso 5 Si no converge se repite el proceso desde el paso 1.

donde † denota la inversa de Moore-Penrose (Schott, 1997).

Por las propiedades del producto de Khatri-Rao es bastante común reescribir las ecuaciones según sigue (Smilde et al., 2004):

$$\mathbf{A} = \mathbf{X}_a(\mathbf{C} \odot \mathbf{B})(\mathbf{C}^T\mathbf{C} * \mathbf{B}^T\mathbf{B})^\dagger$$

donde $*$ es el producto de Hadamard. La ventaja de esta versión es que en esta ecuación es necesario calcular la pseudoinversa de una matriz $R \times R$ en lugar de una matriz $JK \times R$.

Las matrices $\mathbf{B}$ y $\mathbf{C}$ se pueden inicializar de distintas formas: aleatoriamente, tomando los R primeros vectores singulares de la matriz correspondiente ($\mathbf{X}_b$ para $\mathbf{B}$ y $\mathbf{X}_c$ para $\mathbf{C}$), etc (Bro, 1988; Smilde et al., 2004).

2.5.2. MODELOS *TUCKER*

Recordemos que el modelo $Tucker3$ es de la forma:

$$x_{ijk} = \sum_{p=1}^{P}\sum_{q=1}^{Q}\sum_{r=1}^{R} a_{ip}b_{jq}c_{kr}g_{pqr} + e_{ijk} \tag{2.43}$$

Tucker (1966) ofrece una solución para las matrices **A**, **B** y **C** del modelo. Sin embargo en su trabajo plantea que las soluciones encontradas no son estimadores mínimo cuadráticos. Es decir, a pesar de que para rango completo (tomando P = cantidad de componentes subyacentes en el primer modo, Q = cantidad de componentes subyacentes en el segundo modo y R = cantidad de componentes subyacentes en el tercer modo) se logra reproducir el valor x_{ijk}, al retener las primeras componentes de cada modo, el ajuste producido por el modelo puede ser lo suficientemente distante del verdadero valor de x_{ijk} como para ser considerado un mal ajuste.

Para tratar de resolver esta situación Kroonenberg and de Leeuw (1980) proponen un algoritmo (**TUCKALS3**) que minimiza la suma de cuadrados de los términos de error. Especificamente el método minimiza:

$$f(\mathbf{A},\mathbf{B},\mathbf{C},\underline{\mathbf{G}}) = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}(x_{ijk}-\widehat{x}_{ijk})^2 = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}(x_{ijk}-\sum_{p=1}^{P}\sum_{q=1}^{Q}\sum_{r=1}^{R} a_{ip}b_{jq}c_{kr}g_{pqr})^2 \tag{2.44}$$

sobre **A**,**B**,**C** y $\underline{\mathbf{G}}$. Se puede comprobar (Kroonenberg, 1983) que en este caso el ajuste es igual a la suma de los cuadrados de los datos aproximados, es decir, la suma de los cuadrados de $\widehat{x}_{ijk} = \sum_{p=1}^{P}\sum_{q=1}^{Q}\sum_{r=1}^{R} a_{ip}b_{jq}c_{kr}q_{pqr}$. El ajuste se divide habitualmente por la suma de los cuadrados de los datos obteniéndose el porcentaje de ajuste: $\sum_{p=1}^{P}\sum_{q=1}^{Q}\sum_{r=1}^{R}\widehat{x}_{ijk}^2 / \sum_{p=1}^{P}\sum_{q=1}^{Q}\sum_{r=1}^{R} x_{ijk}^2$. Por razones de simplicidad **A**, **B** y **C** se suelen tomar ortogonales por columnas sin perdida de generalidad.

Posteriormente Ten Berge et al. (1987) dan algunos resultados adicionales para el ajuste de los modelos de $Tucker$ relajando condiciones de ortogonalidad.

El algoritmo TUCKALS3 tiene los siguientes pasos:

Paso 0 Se inicializan las matrices **B** y **C** como sigue:
$[\mathbf{B}, S, V] = svd(\mathbf{X}_b, Q)$
$[\mathbf{C}, S, V] = svd(\mathbf{X}_c, R)$

Paso 1 $[\mathbf{A}, S, V] = svd(\mathbf{X}_a(\mathbf{C} \otimes \mathbf{B}), P)$

Paso 2 $[\mathbf{B}, S, V] = svd(\mathbf{X}_b(\mathbf{C} \otimes \mathbf{A}), Q)$

Paso 3 $[\mathbf{C}, S, V] = svd(\mathbf{X}_c(\mathbf{B} \otimes \mathbf{A}), R)$

Paso 4 Se comprueba el criterio de convergencia.

Paso 5 Si no converge se repite el proceso desde el paso 1.

Paso 6 $\underline{\mathbf{G}} = \mathbf{A}'\underline{\mathbf{X}}(\mathbf{C} \otimes \mathbf{B})$

Donde **A** $(I \times P)$, **B** $(J \times Q)$ y **C** $(K \times R)$ son las matrices de componentes, $\underline{\mathbf{G}}$ $(P \times Q \times R)$ es la core matrix y $\otimes$ es el producto de Kronecker.

X_a, X_b, X_c son las matriciaciones del modo A, B y C respectivamente de la matriz de tres vías.

La expresión $[\mathbf{U}, S, V] = svd(\mathbf{X}, P)$ se corresponde con la descomposición SVD de **X** donde se retienen las P primeras componentes. La matriz **U** contiene los P primeros vectores singulares izquierdos, **V** los P primeros vectores singulares derechos y **S** es la matriz diagonal que contiene los P primeros valores singulares.

El modelo $Tucker2$ es de la forma:

$$x_{ijk} = \sum_{p=1}^{P} \sum_{q=1}^{Q} a_{ip} b_{jq} g_{pqk} + e_{ijk} \tag{2.45}$$

El algortmo TUCKALS2 (Kroonenberg and de Leeuw, 1980) permite aproximar los valores de las matrices **A**, **B** y $\underline{\mathbf{G}}$ como sigue:

Paso 0 Se inicializan las matrices **B** y **C** como sigue:
$[\mathbf{B}, S, V] = svd(\mathbf{X}_b, Q)$
C es la matriz identidad de orden R x R

Paso 1 $[\mathbf{A}, S, V] = svd(\mathbf{X}_a(\mathbf{C} \otimes \mathbf{B}), P)$

Paso 2 $[\mathbf{B}, S, V] = svd(\mathbf{X}_b(\mathbf{C} \otimes \mathbf{A}), Q)$

Paso 3 Se comprueba el criterio de convergencia.

Paso 4 Si no converge se repite el proceso desde el paso 1.

Paso 5 $\underline{\mathbf{G}} = \mathbf{A}'\underline{\mathbf{X}}(\mathbf{C} \otimes \mathbf{B})$

2.6. ELECCIÓN DEL NÚMERO DE COMPONENTES A RETENER

2.6.1. MÉTODO DIFFIT

Timmerman and Kiers (2000) proponen un método que se conoce como DIFFIT (DIFFerence in FIT) para seleccionar el número de componentes que se deben retener en el análisis TUCKALS3. Este método ayuda a encontrar un balance óptimo entre el número de componentes retenidas en cada modo y la variabilidad explicada por el modelo. Timmerman and Kiers (2000) plantean que si el número de ejes o componentes han sido elegidos adecuadamente, rara vez el algoritmo TUCKALS3 conduce a un óptimo local.

En el método DIFFIT se calculan los valores de ajuste para todas las soluciones posibles obtenidas a partir del algoritmo TUCKALS3. Sólo deben tenerse en cuenta las combinaciones donde $PQ \geq R$, $PR \geq Q$ y $QR \geq P$ ya que el ajuste de un modelo en el que $R > PQ$ es el mismo que el de un modelo donde $R = PQ$ (T. Wansbeek and Verhees, 1989). También se pueden omitir los modelos en los que $P > max(I, JK)$, $Q > max(J, IK)$ y/o $R > max(K, IJ)$ porque el ajuste de estos modelos no es mejor que el de aquellos donde P, Q y R son iguales a $max(I, JK), max(J, IK), max(K, IJ)$, respectivamente. Denotaremos por $S_{max} = max(I, JK) + max(J, IK) + max(K, IJ)$.

A continuación para cada valor de $S = P + Q + R$ se selecciona la combinación con mejor ajuste y se pasa al cálculo de las diferencias de ajuste **(DifFit)** como la diferencia entre la mejor solución con s componentes y la de s - 1 componentes.

Las diferencias de ajuste juegan un papel similar al que juegan los valores propios en el A.C.P de dos vías, los cuales estaban ordenados de forma decreciente, de manera que la ganancia en ajuste de una solución a otra iba decreciendo.

El siguiente paso es quedarnos sólo con el conjunto de soluciones que verifican que todas las soluciones posteriores tienen asociada una diferencia de ajuste menor. Es decir las soluciones que cumplen la condición: $DifFit_s > DifFit_{s+n}, n = 1, ..., S_{max} - S$.

Para las soluciones retenidas se calculan los cocientes $b_m = \frac{dif_m}{dif_{m+1}}$. Un valor grande de b_m indica que la inclusión de m componentes en lugar de m - 1 incrementa considerablemente el ajuste del modelo.

La solución óptima será aquella para la cual el valor de b_m sea máximo y la diferencia de ajuste asociada a este valor sea mayor que el valor

$$\frac{\|X\|^2}{(S_{max} - 3)}$$

Una desventaja del procedimiento de Timmerman and Kiers (2000) es el tiempo de cálculo que se consume (Kiers and Kinderen, 2003). Estos autores proponen un método alternativo en el que se calculan valores aproximados de ajuste y se dan soluciones en un sólo paso para todas las combinaciones de numero de componentes posibles. Este método es el propuesto por Tucker (1966) para ajustar el modelo $Tucker3$. A pesar de que este método nos lleva a encontrar soluciones no óptimas Kiers and Kinderen (2003) afirman que el ajuste del modelo no es mucho peor que el que se consigue por el procedimiento de los mínimos cuadrados alternados y que además cuando se aumenta el número de componentes, el incremento en el ajuste según el método de Tucker nos proporciona un buen indicador del incremento en el ajuste según el método de los mínimos cuadrados alternados.

Recordemos el método de Tucker para la estimación de las matrices de componentes y la core matrix (ya que este es el proponen utilizar para el cálculo del ajuste aproximado):

Paso 1. Se realiza la descomposición en valores singulares de la matriz $\mathbf{X}_a$: $\mathbf{X}_a\mathbf{X}_a^{'} = \mathbf{K}_a\lambda_a\mathbf{K}_a^{'}$ y se define $\mathbf{A} = \mathbf{K}_{a(1:P)}$ (es decir se toman las P primeras columnas de $\mathbf{K}_a$)

Paso 2. Se realiza la descomposición en valores singulares de la matriz $\mathbf{X}_b$: $\mathbf{X}_b\mathbf{X}_b^{'} = \mathbf{K}_b\lambda_b\mathbf{K}_b^{'}$ y se define $\mathbf{B} = \mathbf{K}_{b(1:Q)}$ (es decir se toman las Q primeras columnas de $\mathbf{K}_b$)

Paso 3. Se realiza la descomposición en valores singulares de la matriz $\mathbf{X}_c$: $\mathbf{X}_c\mathbf{X}_c^{'} = \mathbf{K}_c\lambda_c\mathbf{K}_c^{'}$ y se define $\mathbf{C} = \mathbf{K}_{c(1:R)}$ (es decir se toman las R primeras columnas de $\mathbf{K}_c$)

Paso 4. Se calcula la core matrix $\mathbf{G}_a = \mathbf{A}^{'}\mathbf{X}_a(\mathbf{C} \otimes \mathbf{B})$

Paso 5. Se calcula el porcentaje de ajuste como la suma de los cuadrados de los elementos de $\underline{\mathbf{G}}$ dividido por la suma de los cuadrados de los elementos de $\underline{\mathbf{X}}$.

De esta forma tenemos un procedimiento que nos permite calcular el ajuste según el método de Tucker. A este ajuste, que no es el ajuste mínimo cuadrático, es al que vamos a denominar ajuste aproximado. Es de esperar que la utilización del ajuste óptimo o del ajuste aproximado no dará lugar a grandes diferencias en el cálculo DIFFIT y sin embargo el ajuste aproximado se puede calcular más facilmente.

Para calcular los valores de ajuste para todos los posibles números de componentes se puede utilizar el siguiente procedimiento. En primer lugar se calcula la core matrix completa $\underline{\mathbf{H}}$ asociada con el máximo número de componentes para los tres modos, esto es $\mathbf{A} = \mathbf{K}_a, \mathbf{B} = \mathbf{K}_b, \mathbf{C} = \mathbf{K}_c, \mathbf{H}_a = \mathbf{K}_a^{'}\mathbf{X}_a(\mathbf{K}_c \otimes \mathbf{K}_b)$. A partir de la matriz $\underline{\mathbf{H}}$ se puede obtener la core matrix $\underline{\mathbf{G}}$ relativa a cualquier número de componentes. Esto es debido a que los elementos de la matriz $\underline{\mathbf{H}}$ son de la forma $b_{pqr} = \alpha_p^{'}X_a(\gamma_r \otimes \beta_q)$, donde α_p, β_q y γ_r se corresponden con la p-ésima columna de $\mathbf{K}_a$, la q-ésima columna de $\mathbf{K}_b$ y la r-ésima columna de $\mathbf{K}_c$ respectivamente, con p = 1,...,I, q = 1,...,J, r = 1,...,K. Por otro lado, para un número dado de componentes (P,Q,R), los elementos de la core matrix $\underline{\mathbf{G}}$ son de la forma $g_{pqr} = a_p^{'}X_a(c_r \otimes b_q)$, donde a_p, b_q y c_r denotan la p-ésima columna de $\mathbf{A}$, la q-ésima columna de $\mathbf{B}$ y la r-ésima columna de $\mathbf{C}$ respectivamente, para p = 1,...,I, q = 1,...,J, r = 1,...,K. Dado que $\mathbf{A}$, $\mathbf{B}$ y $\mathbf{C}$ se han tomado como $\mathbf{K}_{a(1:P)}$, $\mathbf{K}_{b(1:Q)}$ y $\mathbf{K}_{c(1:R)}$ respectivamente, se sigue que $a_p = \alpha_p$, $b_q = \beta_q$ y $c_r = \gamma_r$ y por tanto $g_{pqr} = h_{pqr}$, para p = 1,...,P, q = 1,...,Q, r = 1,...,R. Por tanto se pueden calcular en un solo paso los valores aproximados de ajuste de las soluciones para todos los posibles números de componentes, para ello basta calcular la core matrix completa $\underline{\mathbf{H}}$ y tomar la suma de los cuadrados de los elementos de dicha matriz que correspondan según el número de componentes a retener.

Kiers and Kinderen (2003) proponen utilizar los valores de ajuste aproximados cal-

culados según se ha indicado anteriormente en lugar de los valores de ajuste óptimos y seguir el método DIFFIT propuesto por Timmerman and Kiers (2000).

2.6.2. CONVEX HULL NUMÉRICO

Ceulemans and Kiers (2006) presentan un método numérico basado en la envolvente convexa (convex hull) del gráfico en el que se representa la bondad de ajuste frente al número de parámetros libres. Este método consta de los siguientes pasos:

Paso 1. Calcular los valores f_p (número de parámetros libres) y f (bondad de ajuste) para el modelo de tres vías considerado.

El valor aproximado de la bondad de ajuste para una solución T3 de complejidad (P,Q,R) se calcula dividiendo la suma de los cuadrados de los elementos del subarray $\underline{\mathbf{G}}$ de $\underline{\mathbf{H}}$ que se obtiene reteniendo las primeras P, Q y R componentes entre la suma de los cuadrados de los elementos de $\underline{\mathbf{X}}$.

$$f = \frac{\sum_{p=1}^{P}\sum_{q=1}^{Q}\sum_{r=1}^{R} g_{pqr}^2}{\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K} x_{ijk}^2} \tag{2.46}$$

En cuanto al número de parámetros libres f_p de los distintos modelos de tres vías Weesie and Houwelingen (1983) afirman que el número de parámetros libres de una solución T3 de complejidad (P,Q,R) es igual a $IP+JQ+KR+PQR-P^2-Q^2-R^2$. Por la misma razón el número de parámetros libres f_p de una solución T2 de complejidad (Q,R) es igual a $JQ + KR + IQR - Q^2 - R^2$ y el de una solución T1 es igual a $IP + PJK - P^2$. En el caso de la solución CP el número de parámetros libres es igual a $(I + J + K)R - 2R$.

Paso 2. Para cada uno de los n valores f_p observados se retiene sólo la solución con mejor ajuste (con esto se reduce el número de soluciones que deben ser comparadas).

Paso 3. Se ordenan las n soluciones retenidas según el valor de f_p y se denotan como s_i (i = 1,...,n).

Paso 4. Se excluyen las soluciones s_i para las que exista una solución s_j (j ¡i) con $f_j > f_i$. Este paso implica que la linea que une los puntos solución no puede ser decreciente.

Paso 5. Se tienen en cuenta consecutivamente todos los tripletes de soluciones adyacentes. Se excluye la solución intermedia si su punto está localizado por debajo o

sobre la linea que une los puntos vecinos en el gráfico en el que se representa la bondad de ajuste frente al número de parámetros libres.

Paso 6. Se repite el paso 5 hasta que no se puedan excluir más soluciones.

Paso 7. Se calculan los valores s_t para las soluciones retenidas:

$$st_i = \frac{f_i - f_{i-1}}{fp_i - fp_{i-1}} / \frac{f_{i+1} - f_i}{fp_{i+1} - fp_i} \tag{2.47}$$

Paso 8. Se selecciona la solución con el mayor valor de s_t

2.6.3. MÉTODO CHULL

El método CHull fue propuesto por Wilderjans et al. (2013). Originalmente nace dentro del análisis multi-vía, sin embargo Wilderjans et al. (2013) demuestran como este método se puede usar en el contexto del PCA (Pearson, 1901; Hotelling, 1933), reduced-K-means (RKM) (Timmerman et al., 2010), y en modelos de regresión.

El objetivo del método CHull es encontrar un modelo para el que exista un buen balance entre la bondad de ajuste/desajuste y la complejidad del modelo.

Para aplicar el método CHull es necesario ajustar un conjunto de modelos y calcular para cada uno de ellos un valor f de bondad de ajuste/desajuste (e.g., la suma de los cuadrados de los residuales o la cantidad de variabilidad explicada) y un valor de complejidad c. Para medir la complejidad existen varias posibilidades como son el número de parámetros efectivos, el número de factores/componentes (en el contexto del análisis factorial y de componentes principales), o el número de clusters (en el contexto del análisis cluster).

Los pasos del método CHull son los siguientes:

1. Para cada nivel de complejidad c, nos quedamos solo con el mejor modelo (i.e., el modelo con mayor bondad de ajuste o menor desajuste). Cuando haya varios modelos con la misma bondad de ajuste o de desajuste se selecciona uno de ellos de forma aleatoria.

2. Se ordenan los modelos resultantes del paso anterior $m_i (i = 1, ..., n)$ según su complejidad c_i, de manera que el modelo más simple será el primero y el más complejo el último.

3. Se consideran todos los pares de modelos adyacentes y se excluyen aquellos modelos m_i para los que $f_j \geq f_i$ (en el caso de la bondad de ajuste) o $f_j \leq f_i$ (en caso de desajuste) con $j < i$. Repetiremos este paso hasta que no podamos excluir ningún modelo. Por tanto, los valores f_i resultantes seran estrictamente crecientes (en el caso de la bondad de ajuste) o estrictamente decrecientes (en el caso del desajuste).

4. Para cada terna de modelos adyacentes (m_i, m_j, m_k) se eliminan aquellos m_j para los que se verifique $f_j \leq f_i + (c_j - c_i)\frac{(f_k - f_i)}{c_k - c_i}$ (en el caso de la bondad de ajuste) o $f_j \geq f_i + (c_j - c_i)\frac{(f_k - f_i)}{c_k - c_i}$ (en el caso del desajuste). Por tanto se eliminan los modelos que están localizados sobre o debajo (en el caso de la bondad de ajuste) o encima (en el caso del desajuste) de la línea que conecta los otros dos modelos (i.e., m_i y m_k). Se repite este paso hasta que no se eliminen más modelos.

5. Para cada modelo se calcula el valor *st*: $st_i = \frac{\frac{f_i - f_{i-1}}{c_i - c_{i-1}}}{\frac{f_{i+1} - f_i}{c_{i+1} - c_i}}$

6. Se retiene el modelo que tenga el mayor valor *st*. En caso de que haya dos modelos que tengan el mismo valor nos quedaremos con el más simple (i.e., aquel con menor valor c_i).

2.6.4. CORCONDIA

CORCONDIA *(core consistency diagnostic)*, propuesto por Bro and Kiers (2003) es un método para determinar el número de componentes para modelos $PARAFAC$. Este método selecciona el modelo $PARAFAC$ más complejo que es apropiado. Apropiado en el sentido de que si se omite la restricción propia de los modelos $PARAFAC$ que consiste en que la core matrix $\mathbf{G}$ es superdiagonal y se permiten por tanto interacciones entre los componentes de los distintos modos, esto no incrementa significativamente el ajuste del modelo. Por tanto dadas las matrices de componentes $\mathbf{A}$, $\mathbf{B}$ y $\mathbf{C}$ obtenidas mediante la aplicación del algoritmo $PARAFAC$, CORCONDIA compara los elementos de una core matrix sin restricciones con la core matrix superdiagonal $\mathbf{T}$ (i.e. $t_{pqr} = 1$ si $p = q = r$ y $t_{pqr} = 0$ en otro caso):

$$core - consistency = 100.(1 - \frac{\sum_{r_1}^{R}\sum_{r_2}^{R}\sum_{r_3}^{R}(g_{r_1r_2r_3}^{T3} - t_{r_1r_2r_3})^2}{R}) \tag{2.48}$$

donde G^{T3} se obtiene como:

$$vecG^{T3} = (C \odot B \otimes A)^{\dagger} vecX \tag{2.49}$$

donde '†' representa la matriz pseudoinversa. Un valor para la core consistency próximo al 100 por cien indica que la core matrix del modelo $Tucker3$ tiene casi la forma de la matriz superidentidad y por tanto que la solución $PARAFAC$ es apropiada. Un valor para la core consistency del 50 por ciento o inferior significa que los elementos de la core matrix del modelo $Tucker3$ que no están en la diagonal son bastante grandes y por tanto la solución $PARAFAC$ no es apropiada.

Bro and Kiers (2003) recomiendan elegir el modelo $PARAFAC$ más complejo que a su vez verifique que si se añade una componente más al modelo baja drasticamente la core consistency.

Ceulemans and Kiers (2009) proponen formalizar el método CORCONDIA como sigue:

Paso 1 Determinar el valor de la core consistency cc para todas las soluciones $PARAFAC$ entre las que se quiera elegir.

Paso 2 Para cada solución s_R, donde R indica el número de componentes consideradas, se determina el valor de d_R que cuantifica cómo se decrementa el valor de la core consistency cuando se añade una componente:

$$d_R = \frac{cc_R - cc_{R+1}}{cc_{R-1} - cc_R} \tag{2.50}$$

Paso 3 Se selecciona la solución s_R que tenga el mayor valor d_R de entre las que tienen $cc_R \geq 80$. El valor de corte de 80 fue obtenido por Bro and Kiers (2003) a partir de pruebas con un conjunto de datos reales.

2.7. INTERPRETACIÓN DE LA CORE MATRIX

Vamos a ver cómo se pueden interpretar los elementos de la core matrix de los modelos $Tucker3$ (T3) y $Tucker2$ (T2).

La core matrix indica cómo se relacionan las componentes de los tres modos. Así el elemento g_{111} de la core matrix (Figura 2.10) del modelo T3 indica la relación entre la primera componente de los tres modos, el elemento g_{221} la relación entre la segunda componente de los modos 1 y 2 con la primera componente del tercer modo, en general el valor g_{pqr} se considera como una medida de la relación entre la componente p del primer modo, la componente q del segundo modo y la componente r del tercer modo. Por tanto la cantidad:

$$\frac{g_{pqr}^2}{\sum_{pqr} g_{pqr}^2} \tag{2.51}$$

representa la parte de la variabilidad de los datos explicada por el análisis que es atribuida a esa combinación de componentes.

De igual forma si queremos conocer qué variabilidad absorbe cada componente en particular basta con sumar los valores de g_{pqr}^2 manteniendo fijo el índice asociado a la componente analizada. Así por ejemplo si queremos conocer la importancia de la componente q del segundo modo calcularemos:

$$\frac{\sum_{p=1}^{P}\sum_{r=1}^{R} g_{pqr}^2}{\sum_{p=1}^{P}\sum_{q=1}^{Q}\sum_{r=1}^{R} g_{pqr}^2} \tag{2.52}$$

La interpretación de la core matrix del modelo T3 es compleja, por un lado por el signo de sus elementos y por otro lado por el hecho de que hablamos de variables continuas (Kroonenberg, 1983).

Supongamos que el **elemento g_{pqr} es positivo**, es decir que la interacción de la p-ésima, q-ésima y r-ésima componente del primer, segundo y tercer modo respectivamente

.

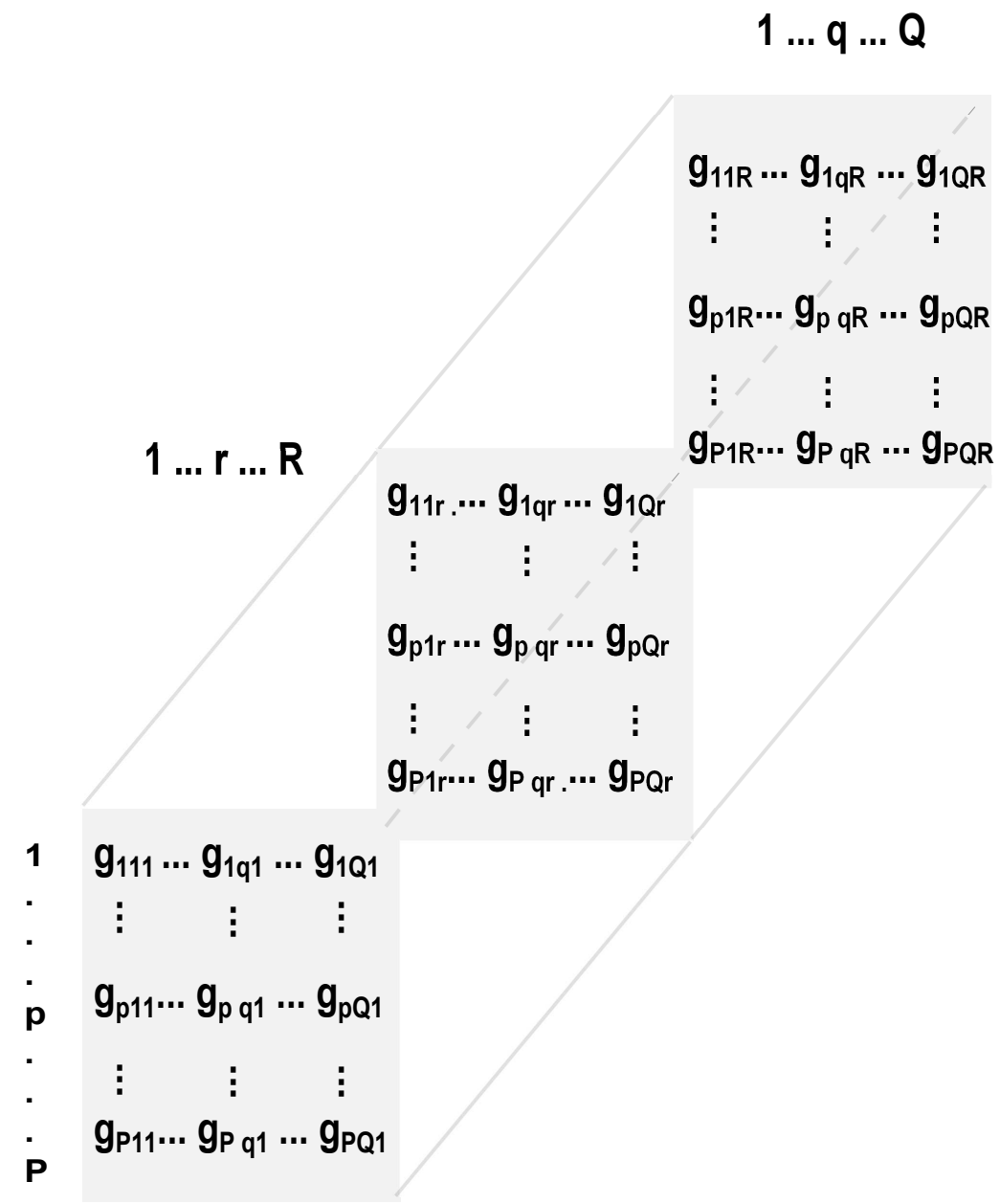


Figura 2.10: Notación de la T3 core matrix .

es positiva. En este caso son posibles cuatro combinaciones de elementos de los tres modos:

$$(+\ ,\ +\ ,\ +)\ (+\ ,\ -\ ,\ -)\ (-\ ,\ +\ ,\ -)\ (-\ ,\ -\ ,\ +)$$

Cada una de estas combinaciones tiene el siguiente significado:

- (+ , + , +): Significa que categorías de i con valores altos en a_{ip}, tienden a tener altos valores de la variable dependiente para combinaciones de categorías jk con altos pesos en b_{jq} y c_{kr}.
- (+ , - , -): Significa que categorías de i con valores altos en a_{ip}, tienden a tener altos valores de la variable dependiente para combinaciones de categorías jk con bajos pesos en b_{jq} y c_{kr}.
- (- , + , -): Significa que categorías de i con valores bajos en a_{ip}, tienden a tener altos valores de la variable dependiente para combinaciones de categorías jk con altos pesos en b_{jq} y bajos pesos en c_{kr}.
- (- , - , +): Significa que categorías de i con valores bajos en a_{ip}, tienden a tener altos valores de la variable dependiente para combinaciones de categorías jk con bajos pesos en b_{jq} y altos pesos en c_{kr}.

Podemos concluir que todas estas combinaciones de categorías de j y k serán las mejores predictoras para predecir los valores más altos de la categoría i-ésima de la variable respuesta en el análisis, es decir, todas las combinaciones de $(a_{ip}, b_{jq}, c_{kr}) > 0$ y $g_{pqr} > 0$ interactúan positivamente.

En caso contrario, es decir, si el signo de **elemento g_{pqr} es negativo**, son posibles otras cuatro combinaciones asociadas al signo de los pesos en **A**, **B**, **C**:

$$(+\ ,\ -\ ,\ +)\ (+\ ,\ +\ ,\ -)\ (-\ ,\ +\ ,\ +)\ (-\ ,\ -\ ,\ -)$$

- (+ , - , +): Significa que categorías de i con valores altos en a_{ip}, tienden a tener altos valores de la variable dependiente para combinaciones de categorías jk con bajos pesos en b_{jq} y altos pesos en c_{kr}.

- (+ , + , -): Significa que categorías de i con valores altos en a_{ip}, tienden a tener altos valores de la variable dependiente para combinaciones de categorías jk con altos pesos en b_{jq} y bajos pesos en c_{kr}.
- (- , + , +): Significa que categorías de i con valores bajos en a_{ip}, tienden a tener altos valores de la variable dependiente para combinaciones de categorías jk con altos pesos en b_{jq} y altos pesos en c_{kr}.
- (- , - , -): Significa que categorías de i con valores bajos en a_{ip}, tienden a tener altos valores de la variable dependiente para combinaciones de categorías jk con bajos pesos en b_{jq} y bajos pesos en c_{kr}.

Al igual que ocurre en el caso de las dos vías sólo se realiza este análisis con las combinaciones de categorías ijk con mayor valor absoluto en las respectivas matrices de marcadores. Es decir, sólo se trabaja con las categorías que caracterizan cada componente.

2.8. REPRESENTACIÓN GRÁFICA

Carlier and Kroonenberg (1996) proponen dos tipos de representaciones biplot para datos con estructura de tres vías: el biplot interactivo y el biplot conjunto. En ambos casos se van a presentar en el contexto del modelo $Tucker3$ pero se podría aplicar a cualquier otra descomposición en tres vías.

2.8.1. BIPLOT INTERACTIVO

Consiste en combinar dos de los modos, en nuestro caso j y k. Tendremos por tanto marcadores a_i y marcadores h_{jk} de la siguiente manera:

$$x_{ijk} \simeq \sum_{p=1}^{P} a_{ip}(\sum_{q=1}^{Q}\sum_{r=1}^{R} g_{pqr}b_{jq}c_{kr}) = \sum_{p=1}^{P} a_{ip}h_{(jk)p} \tag{2.53}$$

Por tanto el número de ejes utilizados en la representación, P, viene determinado por el número de componentes retenidos en el primer modo.

Este tipo de representación es adecuada cuando uno de los modos que interactúan está ordenado (por ejemplo si el modo es el tiempo), o cuando la cantidad de niveles de los modos que se concatenan no es excesivamente grande (Carlier and Kroonenberg, 1996).

Si suponemos, que k es un modo ordenado, podemos trazar trayectorias en el biplot conectando, para cada j, los puntos (j,k) en su propio orden y facilitará la interpretación, especialmente si k es una vía que representa el tiempo (Carlier and Kroonenberg, 1996).

Si el número de marcadores jk es excesivamente grande parece desaconsejable una representación de este tipo, por lo que se propone otro tipo de representación donde la información quede resumida de manera más simple.

2.8.2. BIPLOT CONJUNTO

En este caso lo que se hace es un biplot condicional a uno de los modos (por ejemplo el modo K). El objetivo es hacer un biplot para cada componente del tercer modo en el que se representan los marcadores asociados a los otros dos modos: a_i y b_j.

Para cada componente r del tercer modo se construye la matriz

$$H_r = AG_rB^{'} \tag{2.54}$$

y se realiza el biplot; es decir la descomposición en valores singulares de H_r (matriz de orden $I \times J$). En este biplot quedan proyectadas las categorías del primer y segundo modo, proyectadas sobre la componente r del tercer modo. En G_r se encuentra la parte de la core matrix $\mathbf{G}$, relacionada con la componente r del tercer modo.

Para relacionar las categorías del tercer modo con las categorías de los dos primeros modos representados en el gráfico, utilizamos los pesos asociados a las categorías del tercer modo en la componente r; contenidos en la matriz $\mathbf{C}$; siendo muy importante el signo de cada peso.

Por ejemplo, si la categoría k del tercer modo tiene asociada un peso negativo alto en la componente r del tercer modo, proximidades entre los marcadores a_i y b_j en el gráfico, se interpretan como que interactúan de manera negativa con la categoria k del tercer modo. De igual modo marcadores a_i y b_j distantes en la representación, indican una interacción positiva con la categoria k del tercer modo.

Cada representación gráfica se relaciona con las categorías del tercer modo con pesos altos (en valor absoluto) en la componente del tercer modo sobre la cual se está condicionando.

2.9. APLICACIONES

Sin pretender ser exhaustivos, vamos a hacer un pequeño repaso de las aplicaciones en los distintos campos de los métodos anteriores:

2.9.1. MODELO $PARAFAC$

El modelo CP nace dentro del ámbito de la **psicometría** en 1970. Harshman (1970) aplica el modelo $PARAFAC$ en datos de sonidos vocalísticos, donde distintos individuos (modo A) pronuncian distintas vocales (modo B) midiéndose el tono (modo C). Appellof and Davidson (1981) fueron los pioneros en el uso del modelo CP en **quimiometría**. Andersen and Bro (2003) extienden el uso del modelo CP en datos de quimiometría, en particular el modelo CP ha sido muy útil en el modelado de datos de excitación-emisión de fluorescencias.

Sidiropoulos et al. (2000a) utilizan el modelo CP en el procesamiento de arrays de sensores. Otras aplicaciones en **telecomunicaciones** se deben a Sidiropoulos et al. (2000b), Sidiropoulos and Budampati (2002) y Lathauwer and Castaing (2007).

Varios autores han utilizado la descomposición CP en el ámbito de la **neurociencia**. Mocks (1988) utiliza el modelo CP con imagenes cerebrales, Andersen and Rayens (2004) utilizan el modelo CP con datos FMRI (Imagen por Resonancia Magnética Funcional), Martinez-Montes et al. (2004) y Miwakeichi et al. (2004) utilizan CP con datos de encefalogramas. Más recientemente Acar et al. (2007) y Vos et al. (2007a,b) utilizan el modelo CP para analizar las crisis epilépticas.

La primera aplicación del modelo CP en **minería de datos** se debe a Acar et al. (2005, 2006) los cuales aplican distintan descomposiciones CP a datos de conversaciones por chat. En el análisis de textos, Bader et al. (2007) usan el modelo CP para el análisis de conversaciones vía correo electrónico.

Shashua and Levin (2001) aplican el modelo CP en la clasificación y **compresión de imágenes**. Furukawa et al. (2002) aplica el modelo CP a las funciones BTF (bidirectional texture functions) con el fin de construir una base de datos comprimida de texturas. Bauckhage (2007) extienden el análisis para datos de orden superior (imagenes en color).

Algunos trabajos recientes han utilizado el modelo CP para la resolución de ecuaciones en derivadas parciales estocásticas (Zander and Matthies, 2007; Doostan et al., 2007).

2.9.2. MODELOS *TUCKER*

Existen varios ejemplos de utilización de la descomposición *Tucker* en el ámbito de la **química** propuestos por Henrion (1994) como parte de su tutorial sobre *PCA* de N-vías. Kiers and Mechelen (2001) proponen ejemplos en el ámbito de la **psicometría** dentro del repaso que realizan de las técnicas de análisis de datos de tres vías. Lathauwer and Vandewalle (2004) aplican la descomposición *Tucker* al procesamiento de señales. Muti and Bourennane (2005) aplican la descomposición de *Tucker* para extender los filtros de Wiener en el procesamiento de la señal.

Vasilescu and Terzopoulos (2002) son pioneros en la utilización de la descomposición de *Tucker* en el **análisis de imagenes** por computador (TensorFaces). Consideran datos de imagenes faciales donde para cada individuo tienen múltiples fotografías tomadas en distintas condiciones. Por ejemplo, si la variación es la iluminación los datos se organizan en tres modos: individuo, condiciones de iluminación y píxeles. TensorFaces permite eliminar los efectos irrelevantes, como la iluminación, reteniéndose unicamente los rasgos faciales importantes (Vasilescu and Terzopoulos, 2003). Vasilescu (2002) también utiliza la descomposición de *Tucker* con el **movimiento humano**.

Wang and Ahuja (2003) utilizan el modelo de *Tucker* para modelar **expresiones faciales** y para la compresión de datos de imagenes, Vlasic et al. (2005) lo utilizan para transferir expresiones faciales. Existen muchas aplicaciones dentro del procesamiento de imagenes, como son las de Nagy and Kilmer (2006), Vasilescu and Terzopoulos (2004) y Abdallah et al. (2007).

La descomposición *Tucker* también ha sido utilizada en **data mining** por diferentes autores: Savas and Elden (2007), Acar et al. (2005), Acar et al. (2006), Sun et al. (2006)

3

ANÁLISIS DE DATOS DE TIPO BINARIO EN TRES VÍAS

3.1. INTRODUCCIÓN

En este capítulo se presentan un conjunto de modelos válidos para **datos binarios** en tres vías.

Este tipo de datos son muy frecuentes en psicología. Un ejemplo podría ser el siguiente: Se pasa el cuestionario Child Behavior Checklist (CBCL) a 73 chicos en cuatro momentos diferentes dicotomizándose las respuestas a cada uno de los ítems del cuestionario (0-no ocurre, 1-ocurre). Por tanto tendríamos una matriz de datos binarios del tipo (Chicos $\times$ Items $\times$ Tiempo) de tamaño $(73 \times 108 \times 4)$ (Figura 3.1).

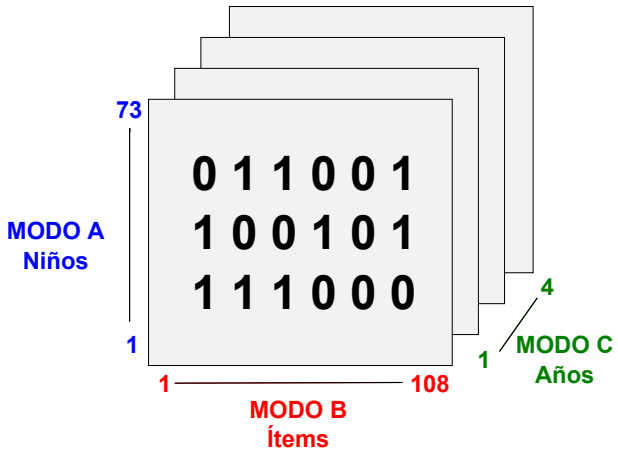


Figura 3.1: Ejemplo de tres vías con datos binarios.

Según hemos visto en el Capítulo 2 para la modelización de datos de tipo continuo en tres vías se utiliza una familia de modelos que generalizan el PCA; esta familia de modelos incluyen los modelos $PARAFAC/CANDECOMP$, $Tucker3$ y $Tucker2$.

Para la representación de datos binarios en N-vías podemos considerar la familia de **modelos de clases jerárquicas** ($HICLAS$) (De Boeck and Rosenberg (1988); Van Mechelen et al. (1995); Leenen et al. (1999, 2001)). Estos modelos no sólo incluyen

una regla de descomposición sino que también representan relaciones de equivalencia y jerarquía entre los elementos de cada modo.

En estos modelos nos vamos a referir a los elementos del primer modo como **objetos**, a los del segundo modo como **atributos** y a los del tercer modo como **recursos**.

Dentro de la familia de modelos de clases jerárquicas tenemos el modelo $INDCLAS$ (modelo jerárquico de clases de diferencias individuales) (Leenen et al. (1999)), el modelo $Tucker3$-$HICLAS$ (Ceulemans et al. (2003)), el modelo $Tucker2$-$HICLAS$ (Ceulemans and Van Mechelen (2004)) y el modelo $Tucker1$-$HICLAS$ (Ceulemans and Van Mechelen (2005)). Antes de presentar estos modelos vamos a introducir el modelo $HICLAS$ para tablas de dos vías (De Boeck and Rosenberg, 1988).

3.2. CLASES JERÁRQUICAS

Un modelo *HICLAS* aproxima una matriz de datos binaria **D** de tamaño $(n_1 \times n_2)$ por una matriz binaria **M** de tamaño $(n_1 \times n_2)$ que puede ser descompuesta en una matriz binaria **S** de tamaño $(n_1 \times r)$ y una matriz **P** de tamaño $(n_2 \times r)$, donde r es el rango del modelo. Las filas de **S** (respectivamente **P**) se llaman objetos (respectivamente atributos) y las r columnas de **S** (respectivamente **P**) definen r clusters binarios , llamados paquetes (bundles) de objetos (respectivamente atributos). El patrón paquete (bundle pattern) de un elemento (objeto o atributo) es el conjunto de paquetes a los que pertenece (De Boeck and Rosenberg, 1988).

Para un elemento x de cualquiera de los modos, $M(x)$ es el conjunto de elementos del otro modo con los que está asociado x en **M**. Así en la Tabla 3.1 (Leenen et al., 1999) $M(e) = \{3, 5, 7\}$.

	Atributos				
	a	b	c	d	e
Objetos					
1	0	1	1	1	0
2	0	1	1	1	0
3	0	1	1	0	1
4	1	0	0	1	0
5	0	1	1	0	1
6	0	0	0	0	0
7	1	1	1	1	1

Tabla 3.1: Matriz Binaria

Un modelo *HICLAS* representa tres tipos de relaciones estructurales en **M**:

- Asociación: Es una relación binaria entre el conjunto de objetos y el conjunto de atributos que queda definida por valores 1 en la matriz **M**. El modelo *HICLAS*

	Paquetes de objetos		
Objetos	I	II	III
1	0	1	0
2	0	1	0
3	0	0	1
4	1	0	0
5	0	0	1
6	0	0	0
7	1	1	1

	Paquetes de atributos		
Atributos	I	II	III
a	1	0	0
b	0	1	1
c	0	1	1
d	1	1	0
e	0	0	1

Tabla 3.2: Modelo $HICLAS$ para los datos de la Tabla 3.1 (Leenen et al., 1999)

disyuntivo representa la relación de asociación mediante la siguiente regla:

$$m_{ij} = \bigoplus_{b=1}^{r} s_{ib} p_{jb} \tag{3.1}$$

con $i = 1, ..., n_1$ y $j = 1, ..., n_2$ y $\bigoplus$ la suma Booleana.

Por ejemplo, en el modelo de la Tabla 3.1, el Objeto 3 está asociado con el Atributo b porque ambos elementos pertenecen al paquete III.

$$m_{32} = \bigoplus_{b=1}^{3} s_{3b} p_{2b} \tag{3.2}$$

- Equivalencia: Se define una relación de equivalencia entre los elementos de cada modo. Dos elementos x e y son equivalentes si y solo si M(x) = M(y). Los conjuntos de objetos (atributos) equivalentes se llaman clases de objetos (atributos). En el modelo $HICLAS$ los elementos equivalentes tienen los mismos patrones paquete. En el ejemplo anterior los Objetos 3 y 5 son equivalentes, también lo son los atributos b y c.

- Jerarquía: Las relaciones de jerarquia se definen entre los elementos (o clases) de cada modo. Un elemento (clase) está jerarquicamente por debajo de otro elemento (clase) si y solo si M(x) es un subconjunto de M(y). En el modelo $HICLAS$ disyuntivo es equivalente a decir que el patrón de x es un subconjunto del patrón de y.

En el ejemplo el Objeto 1 está jerarquicamente por debajo del Objeto 7.

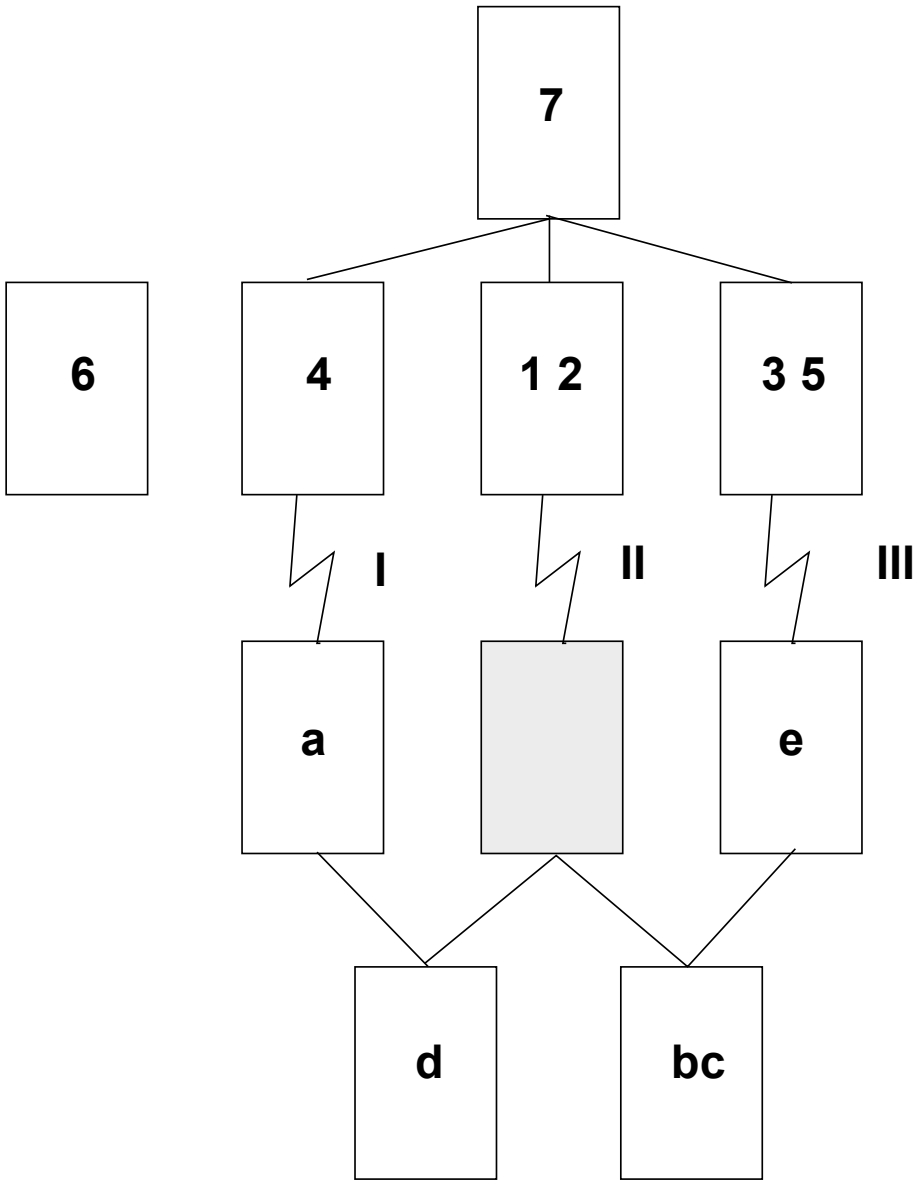


Figura 3.2: Representación Gráfica del Modelo $HICLAS$ de la Tabla 3.2 (Leenen et al., 1999).

De Boeck and Rosenberg (1988) propusieron una representación gráfica para el mo-

delo disyuntivo $HICLAS$. Los objetos se representan en la parte superior y los atributos en la parte inferior. La relación de asociación entre objetos y atributos se representa mediante lineas en zigzag. Los elementos equivalentes se encierran en una caja que representa una clase y las relaciones jerárquicas entre clases se indican mediante lineas que conectan esas clases (Figura 3.2).

A partir de la Figura 3.2 podemos ver que los Objetos 1 y 2 son equivalentes, como también lo son los Objetos 3 y 5. En cuanto a los atributos son equivalentes b y c. El Objeto 7 está jerárquicamente por encima de todos los demás Objetos. El Objeto 4 está asociados con los atributos a y d, los Objetos 1 y 2 con los atributos b, c y d, los Objetos 3 y 5 con los atributos b, c y e y finalmente el Objeto 7 con todos los atributos.

Para el análisis $HICLAS$ existe el programa *HiClas-newGui* desarrollado en la Universidad de Leuven (Bélgica) por la Dra. Eva Ceulemans `http://ppw.kuleuven.be/okp/people/Eva_Ceulemans/`.

3.3. MODELO *INDCLAS*

El modelo *INDCLAS* (Leenen et al., 1999) aproxima un matriz de datos binarios en tres vías **D** de tamaño I (objetos) x J (atributos) x K (recursos) mediante una matriz binaria **M** ($I \times J \times K$) que puede ser descompuesta en una matriz binaria **A** ($I \times R$), una matriz binaria **B** ($J \times R$) y una matriz binaria **C** ($K \times R$), donde R denota el rango del modelo. Las R columnas de **A**, **B** y **C** definen R clusters, llamados paquetes ("bundles") de objetos, atributos y recursos, respectivamente.

En la Tabla 3.3 se muestra un ejemplo de matriz binaria **M** (Leenen et al., 1999).

Recurso A						Recurso B						Recurso C					
	Atributos						Atributos						Atributos					
Objetos	a	b	c	d	e	Objetos	a	b	c	d	e	Objetos	a	b	c	d	e
1	0	1	1	1	0	1	0	1	1	1	0	1	0	1	1	1	0
2	0	1	1	1	0	2	1	1	1	1	0	2	1	1	1	1	0
3	0	1	1	1	0	3	1	1	1	1	0	3	1	1	1	1	0
4	0	0	0	0	0	4	1	0	0	1	0	4	1	0	0	1	0
5	0	1	1	1	1	5	0	1	1	1	1	5	0	1	1	1	0
6	0	0	0	1	1	6	0	0	0	1	1	6	0	0	0	0	0
7	0	0	0	0	0	7	0	0	0	0	0	7	0	0	0	0	0

Tabla 3.3: Matriz de datos binarios

Como cualquier modelo jerárquico de clases, el modelo *INDCLAS* representa tres tipos de relaciones estructurales en **M**: equivalencia, jerarquía y asociación.

Equivalencia Se definen tres relaciones de equivalencia: una en cada modo de **M**. Para el caso de los objetos, el objeto i es equivalente al objeto i' en **M** si y solo si $m_{ijk} = m_{i'jk}$ para todo $j = 1...J, k = 1...K$. Las relaciones entre atributos y recursos se definen de forma análoga. Los objetos equivalentes (respectivamente atributos y recursos) constituyen una clase de objetos (respectivamente atributos y recursos), de manera que estas clases constituyen una partición de los objetos

	Paquetes de objetos				Paquetes de atributos				Paquetes de recursos		
Objetos	I	II	III	Atributos	I	II	III	Recursos	I	II	III
1	0	1	0	a	1	0	0	A	0	1	1
2	1	1	0	b	0	1	0	B	1	1	1
3	1	1	0	c	0	1	0	C	1	1	0
4	1	0	0	d	1	1	1				
5	0	1	1	e	0	0	1				
6	0	0	1								
7	0	0	0								

Tabla 3.4: Modelo $INDCLAS$ para los datos de la Tabla 3.3 (Leenen et al., 1999)

(respectivamente atributos y recursos). El modelo $INDCLAS$ para $\underline{\mathbf{M}}$ representa las relaciones de equivalencia de manera que los elementos equivalentes tengan idénticos patrones paquete.

Por ejemplo en la Tabla 3.3, los objetos 2 y 3 son equivalentes y tienen idénticos patrones paquete en el modelo $INDCLAS$ de la Tabla 3.4.

Jerarquía Se definen relaciones jerárquicas entre los elementos de cada modo de $\underline{\mathbf{M}}$. Con respecto a los objetos, el objeto i está jerarquicamente por debajo del objeto i' en $\underline{\mathbf{M}}$ si y solo si $m_{ijk} \leq m_{i'jk}$ para todo $j = 1...J, k = 1...K$. De igual forma se definen las relaciones jerárquicas entre los atributos y los recursos. Es importante señalar que la definición de relación jerárquica entre los elementos de un modo implica la definición de relación jerárquica entre las clases de ese modo. En el modelo $INDCLAS$ para $\underline{\mathbf{M}}$, las relaciones de jerarquía se representan de la siguiente forma: un elemento (clase) 'x' está jerarquicamente por debajo de un elemento (clase) 'y' si y solo si el patrón paquete de 'x' es un subconjunto del patrón paquete de 'y'. Al conjunto de relaciones de equivalencia y jerárquicas se les llama conjunto de relaciones teóricas.

En la Tabla 3.4, por ejemplo, el Objeto 1 está jerárquicamente por debajo del Objeto 2.

Asociación La relación de asociación es la relación ternaria entre objetos, atributos y recursos que se define como una entrada 1 en la matriz $\underline{\mathbf{M}}$. El modelo $INDCLAS$

representa la relación de asociación mediante la siguiente regla:

$$m_{ijk} = \bigoplus_{r=1}^{R} a_{ir} b_{jr} c_{kr} (\forall i = 1...I, \forall j = 1...J, \forall k = 1...K) \tag{3.3}$$

donde $\bigoplus$ se corresponde con la suma booleana. En resumen, un elemento m_{ijk} es igual a uno si y solo si existe un paquete al que pertenece el objeto i, el atributo j y el recurso k. Por tanto la ecuación 3.3 establece una relación entre las clases jerárquicas de los tres modos que implica una correspondencia uno a uno entre los respectivos paquetes de objetos, atributos y recursos.

De Boeck and Rosenberg (1988) sostienen que para cualesquiera **A**, **B** y **C** descomposición de M según 3.3, existen matrices $\mathbf{A}^*$, $\mathbf{B}^*$ y $\mathbf{C}^*$ del mismo tamaño y que además representan el conjunto de relaciones teóricas.

Leenen et al. (1999) proponen una representación gráfica del modelo $INDCLAS$ que tiene en cuenta los tres tipos de relaciones estructurales del mismo. Para obtener esta representación primero se dibujan las clasificaciones jerárquicas de objetos y atributos en la parte superior e inferior respectivamente. Los elementos equivalentes se representan en cajas que representan las clases equivalentes, y las relaciones de jerarquía entre clases se representan mediante líneas entre las respectivas cajas. Por tanto, las jerarquías entre objetos y atributos se relacionan mediante el dibujo de caminos entre las clases base (es decir, clases de elementos que pertenecen a un sólo paquete) de los paquetes de objetos y atributos. Cada camino incluye un hexágono que contiene las clases de recursos que pertenecen al correspondiente paquete recurso (Figura 3.3).

Las relaciones teóricas entre objetos y atributos quedan perfectamente representadas en este gráfico, mientras que no ocurre lo mismo con las relaciones entre recursos, por lo que se puede hacer una representación separada de la jerarquia entre recursos. A partir de la representación gráfica se puede obtener la siguiente información: un objeto i está asociado con el atributo j por medio del recurso k si y sólo si el objeto i y el atributo j se conectan mediante un camino que pasa por el recurso k.

El modelo $INDCLAS$ es bastante restrictivo por dos motivos. En primer lugar, las

.

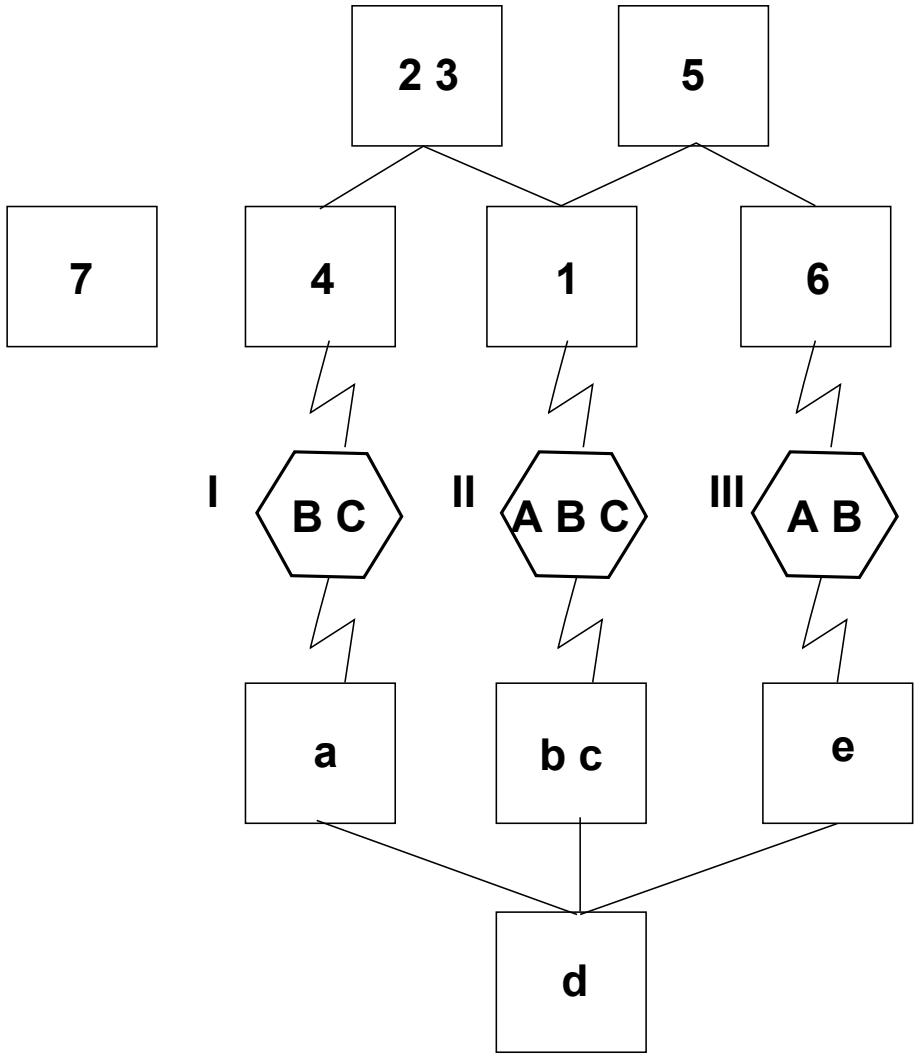


Figura 3.3: Representación gráfica de un modelo *INDCLAS* de rango 3 (Leenen et al., 1999).

clasificaciones jerárquicas de los tres modos tienen que tener el mismo rango, por tanto el número de paquetes tiene que ser el mismo para cada modo. En segundo lugar, la estructura que relaciona las tres jerarquías establece una correspondencia uno a uno entre los respectivos paquetes de objetos, atributos y recursos.

El modelo $INDCLAS$ está muy relacionado con el modelo $PARAFAC/CANDECOMP$ para datos reales en tablas de tres vías (Carroll and Chang (1970); Harshman (1970)). En concreto el modelo $PARAFAC/CANDECOMP$ con componentes binarias difiere del modelo $INDCLAS$ en dos aspectos. En primer lugar, el modelo $INDCLAS$ utiliza una descomposición booleana, mientras que el modelo $PARAFAC/CANDECOMP$ se basa en una descomposición algebraica: si en la ecuación 3.3 se reemplaza $\bigoplus$ por una suma se obtiene el modelo $CANDECOMP/PARAFAC$. En segundo lugar, el modelo $INDCLAS$ restringue los paquetes de matrices para que representen el conjunto de relaciones teóricas.

3.3.1. ALGORITMO

El objetivo del análisis $INDCLAS$ de rango R de un $I \times J \times K$ matriz de datos binarios $\underline{\mathbf{D}}$ es encontrar un $I \times J \times K$ matriz binaria $\underline{\mathbf{M}}$ tal que minimize,

$$\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}(d_{ijk} - m_{ijk})^2 \tag{3.4}$$

sujeto a la restricción de que M pueda ser descompuesta en las matrices paquete $\mathbf{A}$, $\mathbf{B}$ y $\mathbf{C}$ de tamaños $I \times R$, $J \times R$ y $K \times R$ según 3.3 y $\mathbf{A}$, $\mathbf{B}$ y $\mathbf{C}$ verifican las relaciones de equivalencia y jerarquía.

El algorítmo consta de dos subrutinas principales: en la primera subrutina el algorítmo busca las matrices $\mathbf{A}$, $\mathbf{B}$ y $\mathbf{C}$ a partir de las cuales por 3.3 se obtiene la matriz $\underline{\mathbf{M}}$ que hace que el valor de 3.3 sea mínimo. En la segunda subrutina se transforman las matrices anteriores para que $\underline{\mathbf{M}}$ verifique el conjunto de relaciones teóricas.

La primera subrutina se puede considerar como un procedimiento de mínimos cuadrados alternados para datos binarios (Chaturvedi and Carroll (1994)). Comienza a partir de una configuración inicial para dos de las tres matrices paquete. Esta configuración inicial se puede obtener (a) pseudo aleatoriamente (b) racionalmente por medio de algún método incluido en el algorítmo (c) a través del usuario. Asumiendo una configuración inicial $\mathbf{A}^{(0)}$, $\mathbf{B}^{(0)}$ para las matrices $\mathbf{A}$ y $\mathbf{B}$, la matriz paquete óptima $\mathbf{C}^{(0)}$ para la cual 3.4 es mínimo se calcula condicionalmente a $\mathbf{A}^{(0)}$ y $\mathbf{B}^{(0)}$. A continuación se calcula $\mathbf{A}^{(1)}$ a partir de $\mathbf{B}^{(0)}$ y $\mathbf{C}^{(0)}$, después $\mathbf{B}^{(1)}$ condicionado a $\mathbf{A}^{(1)}$ y $\mathbf{C}^{(0)}$, y así sucesivamente. Este proceso finaliza cuando no se observan diferencias en la función de pérdida 3.4.

Para la estimación de una matriz paquete condicionada a las otras dos matrices paquete se puede usar la propiedad de separabilidad de la función de pérdida (Chaturvedi and Carroll (1994)). Esta propiedad dice que la contribución a la función de pérdida del patrón paquete de por ejemplo el objeto i de A se puede separar de la contribución de los patrones paquete del resto de los objetos de A.

$$\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}(d_{ijk}-\bigoplus_{r=1}^{R}a_{ir}b_{jr}c_{kr})^2=\sum_{j=1}^{J}\sum_{k=1}^{K}(d_{1jk}-\bigoplus_{r=1}^{R}a_{1r}b_{jr}c_{kr})^2+...+$$

$$\sum_{j=1}^{J}\sum_{k=1}^{K}(d_{Ijk}-\bigoplus_{r=1}^{R}a_{Ir}b_{jr}c_{kr})^2 \tag{3.5}$$

Por lo tanto la optimización de A se puede conseguir optimizando sucesivamente las filas $a_{i.}$ de cada término de 3.5. Esto mismo se puede utilizar para la optimización de B y C. Para la obtención del patrón paquete que mejor ajusta a un elemento 'x' se utilizará regresión booleana (Leenen and Van Mechelen (1998); Van Mechelen (1988)).

El algorítmo propuesto para la regresión booleana por Leenen and Van Mechelen (1998) consiste en lo siguiente. Dado un conjunto de valores de N observaciones de un criterio binario C y un conjunto P de n predictores binarios $p_1, p_2, ..., p_n$, el objetivo de la regresión booleana es encontrar una combinación disyuntiva (o conjuntiva) de los predictores que minimice la función objetivo. Esta función objetivo penaliza tanto los errores en la predicción como la complejidad de la expresión lógica. Por lo tanto los

parámetros que intervienen en ella son los siguientes:

1. $c_1, ..., c_n$: un parámetro de coste para cada uno de los n predictores.
2. D^-: un parámetro de coste para los **falsos negativos** (una observación para la cual la expresión lógica vale 0 mientras que el valor del criterio es 1).
3. D^+: un parámetro coste para los **falsos positivos** (una observación para la cual la expresión lógica vale 1 mientras que el valor del criterio es 0).

Vamos a denotar el número de falsos negativos por N_{D^-} y el número de falsos positivos por N_{D^+}. Por tanto para una expresión disyuntiva (conjuntiva), definida por un subconjunto W de los predictores, el valor de la función objetivo se define como:

$$f(W) = D^- N_{D^-} + D^+ N_{D^+} + \sum_{p_i \in W} c_i$$

Es importante señalar que a partir de la definición de la función objetivo y de la equivalencias siguientes

$$(p_{i1} \wedge p_{i2} \wedge ... \wedge p_{ik}) \Leftrightarrow C$$

$$(\neg p_{i1} \vee \neg p_{i2} \vee ... \vee \neg p_{ik}) \Leftrightarrow \neg C$$

se sigue que un algorítmo que se use para obtener la combinación disyuntiva óptima se puede utilizar también para obtener la combinación conjuntiva óptima (y viceversa). Para ello basta con aplicar el algorítmo disyuntivo con $\neg C$ como criterio, $\neg P_1, ..., \neg Pn$ como predictoras, D^+ como coste para los falsos negativos y D^- como coste para los falsos positivos. Por este motivo a continuación nos vamos a referir solo al caso disyuntivo.

Para encontrar la combinación disyuntiva óptima se crea un árbol con 2^n nodos que recoge todos los subconjuntos de P. El árbol tiene n+1 niveles, etiquetados de 0 a n, donde el nivel k contiene todos los subconjuntos de P con k elementos. Para cada nodo del árbol se debe definir algún orden $\succ$ en el conjunto de los nodos hijos. Las relaciones entre padres e hijos en el árbol se pueden definir recursivamente de la siguiente forma:

- Los hijos del nodo de nivel 0 (que representa el conjunto vacío) son todos los elementos individuales.

- Sea V el padre de W, entonces los hijos de un nodo W del nivel k (k > 0) son los conjuntos $W \bigcup C^{(i)}, \forall C^{(i)} \succ W$, donde $C^{(i)}$ es un hijo de V.

El algorítmo recorre este árbol mediante un procedimiento recursivo. Este procedimiento comienza en el nodo raíz del árbol y recorre el árbol de manera que aplicado sobre un nodo S del árbol realiza fundamentalmente dos tareas:

1. $f(S)$ se compara con el menor valor de f encontrado hasta el momento; el conjunto de predictores para el que el valor de f es mínimo se denota como M. Por tanto si $f(S) < f(M)$ entonces M se reemplaza por S.

2. Se evaluan los hijos W de S en orden ascendente con respecto a la siguiente función:

$$f^+ = D^+ N_{D^+} + \sum_{p_i \in W} c_i$$

   Esta función f^+ proporciona un límite inferior de f para todas las posibles extensiones de W; ya que los falsos positivos no se pueden corregir añadiendo expresiones disyuntivas. Por lo tanto no se examina los subarboles descendientes de W que verifican que $f^+(W) \geq f(M)$. Si $f^+(W) < f(M)$ entonces el procedimiento recursivo se aplica a W.

En el caso concreto que nos ocupa para ajustar a_i en el i-ésimo término de 3.4, exitirán r variables predictoras, los valores d_{ijk} son los valores de la variable criterio y los productos $b_{jr}c_{kr}$ son los valores de la r-iésima variable predictora. a_{ir} tendrá valor uno cuando r pertenezca al subconjunto de predictoras que predice de forma óptima el criterio y cero en otro caso.

La segunda subrutina modifica las matrices obtenidas en la primera para que verifiquen el conjunto de relaciones teóricas. Para ello se utiliza sucesivamente la operación de cierre sobre cada una de las matrices paquete (Barbut and Monjardet (1970)). Esta operación consiste en cambiar los ceros por unos en las matrices paquete siempre y cuando este cambio no modifique M (y por lo tanto tampoco el valor de función de pérdida). Se puede demostrar que esta operación de cierre es una operación suficiente aunque no necesaria para garantizar el conjunto de relaciones teóricas.

3.4. MODELO *TUCKER3-HICLAS*

El modelo *Tucker*3-*HICLAS* (Ceulemans et al. (2003)) implica una descomposición de una matriz binaria $\underline{\mathbf{M}}$ de tamaño $I \times J \times K$ en una matriz binaria $\mathbf{A}$ de objetos de tamaño $I \times R$, una matriz binaria $\mathbf{B}$ de atributos de tamaño $J \times S$, una matriz binaria $\mathbf{C}$ de recursos de tamaño $K \times T$ y una matriz binaria $\mathbf{G}$ de tamaño $R \times S \times T$ siendo (R, S, T) el rango del modelo. $\mathbf{G}$ define las relaciones de asociación entre los objetos, atributos y recursos; a esta matriz se le llama core matrix.

El conjunto de relaciones teóricas de equivalencia y jerarquía son idénticas para los modelos *INDCLAS* y *Tucker*3-*HICLAS*, sólo difieren en la relación de asociación. En el modelo *Tucker*3-*HICLAS* la relación de asociación se define como sigue:

$$m_{ijk} = \bigoplus_{r=1}^{R} \bigoplus_{s=1}^{S} \bigoplus_{t=1}^{T} a_{ir} b_{js} c_{kt} g_{rst} (\forall i = 1...I, \forall j = 1...J, \forall k = 1...K) \tag{3.6}$$

Es decir m_{ijk} es igual a 1 si el objeto i, el atributo j y el recurso k pertenecen a un paquete objeto, paquete atributo y paquete recurso y estos estan asociados en la core matrix $\underline{\mathbf{G}}$. Si se reemplaza el $\bigoplus$ en la ecuación 3.6 por una suma se obtiene el modelo *Tucker*3 (Tucker (1966)), por lo que el modelo *Tucker*3-*HICLAS* es el modelo de clases jerárquico homólogo al modelo *Tucker*3.

La representación gráfica del modelo *Tucker*3-*HICLAS* es muy similar a la representación del modelo *INDCLAS*. La diferencia fundamental entre ambas representaciones gráficas radica en la forma en que se vinculan las jerarquías de objetos y de atributos: en la representación *Tucker*3-*HICLAS* existe un camino entre cada clase base de la jerarquía de objetos y cada clase base de la jerarquía de atributos que están asociados mediante $\underline{\mathbf{G}}$. Para ello se incluyen en un hexágono todos los recursos que pertenecen al paquete recurso mediante el cual están asociadas (Figura 3.4).

A partir de la definición del modelo y de la representación es fácil ver que este modelo

	Recurso A						Recurso B						Recurso C					
	Atributos						Atributos						Atributos					
Objetos	a	b	c	d	e	Objetos	a	b	c	d	e	Objetos	a	b	c	d	e
1	1	0	0	1	1	1	0	1	1	1	0	1	1	1	1	1	1
2	0	0	0	0	0	2	0	0	0	0	0	2	0	0	0	0	0
3	0	0	0	0	0	3	1	0	0	1	1	3	1	0	0	1	1
4	0	1	1	1	0	4	0	0	0	0	0	4	0	1	1	1	0
5	1	0	0	1	1	5	1	1	1	1	1	5	1	1	1	1	1
6	1	1	1	1	1	6	0	1	1	1	0	6	1	1	1	1	1
7	0	0	0	0	0	7	1	0	0	1	1	7	1	0	0	1	1

Tabla 3.5: Matriz de datos binarios (Ceulemans et al., 2003)

	Paquetes de objetos				Paquetes de atributos			Paquetes de recursos	
Objetos	OB1	OB2	OB3	Atributos	AB1	AB2	Recursos	SB1	SB2
1	0	1	0	a	1	0	A	0	1
2	0	0	0	b	0	1	B	1	0
3	1	0	0	c	0	1	C	1	1
4	0	0	1	d	1	1			
5	1	1	0	e	1	0			
6	0	1	1						
7	1	0	0						

Tabla 3.6: Matrices Paquete del modelo *Tucker*3-*HICLAS* para los datos de la Tabla 3.5 (Ceulemans et al., 2003)

es menos restrictivo que el modelo *INDCLAS*. En primer lugar porque el número de paquetes puede ser diferente para los tres modos. Y en segundo lugar porque la estructura que relaciona las tres jerarquías no tiene porqué tener la forma de correspondencia uno a uno, en concreto se permiten las estructuras representadas en la core matrix $\underline{\mathbf{G}}$.

3.4.1. ALGORITMO

El objetivo del análisis *Tucker*3-*HICLAS* de rango (R,S,T) de una $I \times J \times K$ matriz de datos binarios $\underline{\mathbf{D}}$ es encontrar un $I \times J \times K$ matriz binaria $\underline{\mathbf{M}}$ que por un lado minimice

		Paquetes recurso	
Paquetes de Objetos	Paquetes de Atributos	SB1	SB2
OB1	AB1	1	0
OB1	AB2	0	0
OB2	AB1	0	1
OB2	AB2	1	0
OB3	AB2	0	1

Tabla 3.7: Core matrix del modelo $Tucker3$-$HICLAS$ para los datos de la Tabla 3.5 (Ceulemans et al., 2003)

la función de pérdida

$$f(\mathbf{M}) = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}(d_{ijk} - m_{ijk})^2 \tag{3.7}$$

y por otro pueda ser representado mediante un modelo $Tucker3$-$HICLAS$ de rango (R,S,T). Dado que $\underline{\mathbf{D}}$ y $\underline{\mathbf{M}}$ son binarias se podrán utilizar tanto los mínimos cuadrados como la menor desviación absoluta.

Dado un $I \times J \times K$ matriz de datos $\underline{\mathbf{D}}$ y un rango (R,S,T) el algorítmo $Tucker3$-$HICLAS$ consta de dos subrutinas principales: en la primera subrutina el algorítmo busca las matrices $\mathbf{A}$, $\mathbf{B}$ y $\mathbf{C}$ y la core matrix $\underline{\mathbf{G}}$ que mediante la ecuación 3.6 dan lugar a una $I \times J \times K$ matriz $\underline{\mathbf{M}}$ para el cual el valor de 3.7 es mínimo. La segunda subrutina transforma las matrices $\mathbf{A}$, $\mathbf{B}$ y $\mathbf{C}$ de manera que representen el conjunto de relaciones teóricas correctamente.

La primera rutina comienza dando una configuración inicial a las tres matrices de paquetes. Existen dos posibles configuraciones iniciales: (a) análisis $INDCLAS$ del $I \times J \times K$ matriz de datos $\underline{\mathbf{D}}$ con rangos R, S y T; y (b) análisis $HICLAS$ De Boeck and Rosenberg (1988) sobre las $I \times JK, J \times IK$ and $K \times IJ$ matrices de datos en rango R, S y T respectivamente. En ambos casos obtendremos una matriz de objetos de rango R, una matriz de atributos de rango S y una matriz de recursos de rango T. Asumiendo una configuración inicial $\mathbf{A}^{(0)}$, $\mathbf{B}^{(0)}$ y $\mathbf{C}^{(0)}$ el procedimiento de los mínimos cuadrados alternados calcula condicionalmente a partir de $\mathbf{A}^{(0)}$, $\mathbf{B}^{(0)}$ y $\mathbf{C}^{(0)}$, la core matrix $\underline{\mathbf{G}}^{(0)}$ que minimiza 3.7. En el siguiente paso, $\mathbf{A}^{(w)}$ se recalcula condicionada a $\mathbf{B}^{(w-1)}$, $\mathbf{C}^{(w-1)}$ y $\underline{\mathbf{G}}^{(w-1)}$; $\mathbf{B}^{(w)}$ se recalcula condicionada a $\mathbf{A}^{(w)}$, $\mathbf{C}^{(w-1)}$ y $\underline{\mathbf{G}}^{(w-1)}$; $\mathbf{C}^{(w)}$ se recalcula

.

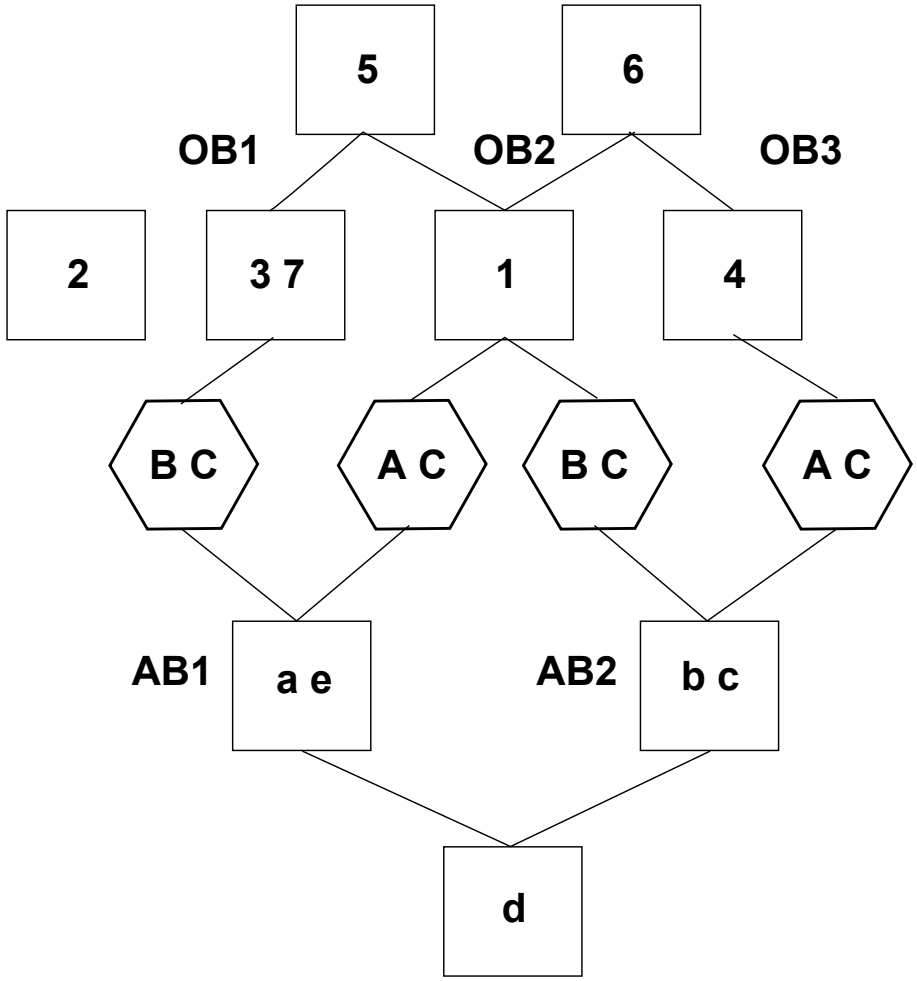


Figura 3.4: Representación gráfica del modelo *Tucker*3-*HICLAS* de la Tabla 3.5 (Ceulemans et al., 2003).

condicionada a $\mathbf{A}^{(w)}$, $\mathbf{B}^{(w)}$ y $\underline{\mathbf{G}}^{(w-1)}$; y $\underline{\mathbf{G}}^{(w)}$ se recalcula condicionada a $\mathbf{A}^{(w)}$, $\mathbf{B}^{(w)}$ y $\mathbf{C}^{(w)}$ (w = 1,2,...). Este proceso finaliza cuando el algorítmo converge.

Sobre cómo realizar la estimación de una matriz paquete condicionada a las otras dos matrices paquete y a la core matrix, el análisis de la función de pérdida muestra que se verifica la propiedad de Chaturvedi and Carroll (1994). Esta propiedad implica que la contribución del patrón paquete de la fila i $a_{i.}$ a la función de pérdida, se puede separar de la contribución de los patrones paquete de las demás filas.

$$\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}(d_{ijk}-\bigoplus_{r=1}^{R}a_{ir}\bigoplus_{s=1}^{S}\bigoplus_{t=1}^{T}b_{js}c_{kt}g_{rst})^2 =$$
$$\sum_{j=1}^{J}\sum_{k=1}^{K}(d_{1jk}-\bigoplus_{r=1}^{R}a_{1r}\bigoplus_{s=1}^{S}\bigoplus_{t=1}^{T}b_{js}c_{kt}g_{rst})^2+...+$$
$$\sum_{j=1}^{J}\sum_{k=1}^{K}(d_{Ijk}-\bigoplus_{r=1}^{R}a_{Ir}\bigoplus_{s=1}^{S}\bigoplus_{t=1}^{T}b_{js}c_{kt}g_{rst})^2$$

Por lo tanto se puede realizar una estimación de **A** (condicionada a **B**, **C** y $\underline{\mathbf{G}}$) mediante optimizaciones sucesivas de los patrones paquete $a_{i.}$ (i = 1...I) de los I términos anteriores. La estimación de **A** mediante filas es mucho más rápida que la estimación de la matriz completa, dado que la estimación por filas implica I evaluaciones de 2^R posibles patrones, mientras que la estimación de la matriz completa implica la evaluación de 2^{IR} patrones. Esta propiedad es válida para la estimación de **B** y **C**, sin embargo la matriz $\underline{\mathbf{G}}$ deberá estimarse completamente.

La estimación de las matrices **A**, **B**, **C** y $\underline{\mathbf{G}}$ se realiza por medio de regresión booleana (Leenen and Van Mechelen, 1998). Por ejemplo, para estimar el patrón paquete $a_{i.}$ del i-ésimo término anterior los valores de la variable criterio son los valores $d_{ijk}(j = 1...J; k = 1...K)$ y los valores de la r-ésima $(r = 1...R)$ variable predictora son las sumas

$$\bigoplus_{s=1}^{S}\bigoplus_{t=1}^{T}b_{js}c_{kt}g_{rst}(j = 1...J; k = 1...K)$$

En la estimación de la core matrix $\underline{\mathbf{G}}$ están involucradas RST variables predictoras; los valores de la variable criterio son los valores $d_{ijk}(i = 1...I; j = 1...J; k = 1...K)$ y los valores de la variable predictora (r, s, t) son los productos $a_{ir}b_{js}c_{kt}(i = 1...I; j = 1...J; k = 1...K)$.

En la segunda rutina se obliga a las matrices paquete obtenidas en la primera rutina a que verifiquen el conjunto de relaciones teóricas. Para ello se aplica una operación de cierre a las matrices **A**, **B** y **C** (Barbut and Monjardet (1970)). Por ejemplo, para hacer que **A** verifique el conjunto de relaciones teoricas entre los objetos cada valor a_{ir} igual a cero se cambia por uno si y solo si para todo j = 1..J, k = 1..K se verifica que

$$\bigoplus_{s=1}^{S}\bigoplus_{t=1}^{T}b_{js}c_{kt}g_{rst} \leq m_{ijk}$$

(lo cual implica que la modificación de **A** no altera $\underline{\mathbf{M}}$).

Señalar que la operación de cierre es una condición suficiente pero no necesaria para garantizar la consistencia del conjunto de relaciones teoricas.

3.4.2. ELECCIÓN DEL NÚMERO DE COMPONENTES

Ceulemans et al. (2003) proponen dos métodos para la selección del número de componentes. El objetivo de ambos es encontrar un modelo con un buen balance entre el ajuste de los datos por un lado y el número de componentes por el otro. El procedimiento es el siguiente:

Paso 1 Se calculan los modelos *Tucker3-HICLAS* $\underline{\mathbf{M}}$ de rangos (1,1,1) hasta (r_1,r_2,r_3) que ajustan a la matriz binaria $\underline{\mathbf{D}}$. Para cada modelo se calcula la suma s de número de objetos, atributos y paquetes, y el BOF (badness-of-fit) según la fórmula:

$$BOF = \frac{\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}(d_{ijk}-m_{ijk})^2}{I\times J\times K}$$

Paso 2 Para cada valor de s se retiene sólo el modelo con mejor ajuste.

Paso 3 Se aplica una variante del scree test (Cattell (1966)). Existen dos variantes para este paso.

Regla A Selecciona el modelo con s maximizando $\frac{BOF_{s^{prev}}-BOF_s}{s-s^{prev}} - \frac{BOF_s-BOF_{s^{next}}}{s^{next}-s}$ siendo s^{prev} y s^{next} los valores anteriores y posteriores de s en el gráfico BOF.

Regla B Selecciona el modelo con s maximizando $\frac{BOF_{s^{min}}-BOF_s}{s-s^{min}} - \frac{BOF_s-BOF_{s^{max}}}{s^{max}-s}$ siendo s^{min} el valor más pequeño de s y s^{max} el mayor valor.

3.5. MODELO *TUCKER2-HICLAS*

La principal diferencia entre el modelo *Tucker3-HICLAS* y el modelo *Tucker2-HICLAS* es que en el modelo *Tucker2* se reducen sólo dos de los tres modos (Ceulemans and Van Mechelen (2004)). Más concretamente un (P,Q) modelo *Tucker2-HICLAS* implica la descomposición de una matriz binaria $\underline{\mathbf{M}}$ en una matriz binaria de objetos $\mathbf{A}$ de tamaño $I \times P$, una matriz binaria de atributos $\mathbf{B}$ de tamaño $J \times Q$ y una matriz core binaria $\underline{\mathbf{G}}$ de tamaño $P \times Q \times K$.

Dado que las relaciones teóricas son idénticas para los modelos *Tucker3* y *Tucker2-HICLAS* sólo se va a discutir sobre la relación de asociación. En el modelo disjuntivo *Tucker2-HICLAS* la relación de asociación se representa mediante la regla:

$$m_{ijk} = \bigoplus_{p=1}^{P} \bigoplus_{q=1}^{Q} a_{ip} b_{jq} g_{pqk} \tag{3.8}$$

ó equivalentemente

$$m_{ijk} = 1 \Longleftrightarrow \exists p, q : a_{ip} = 1 \wedge b_{jq} = 1 \wedge g_{pqk} = 1 \tag{3.9}$$

Existen tres tipos de modelos *Tucker2-HICLAS*: el Tipo A reduce los atributos y los recursos, el Tipo B reduce los objetos y los recursos y el Tipo C reduce los objetos y los atributos.

En la representación gráfica del modelo *Tucker2-HICLAS* disyuntivo la jerarquía de los objetos aparece en la parte superior de la representación. Los objetos equivalentes aparecen dentro de cajas que representan las clases de objetos y la jerarquía entre las clases de objetos se representan mediante líneas que conectan las cajas. De forma similar la jerarquía de los atributos aparece en la parte inferior de la representación. Los objetos y atributos se relacionan mediante caminos entre las clases bases de objetos y atributos según aparecen en $\underline{\mathbf{G}}$. Los hexágonos que aparecen en los caminos indican mediante que recursos se relacionan los objetos y los atributos. En esta representación no se incluyen

las relaciones de dependencia y jerarquía entre los recursos por lo que se puede realizar una representación independiente para la jerarquía de recursos.

En cuanto a la relación entre los modelos disyuntivos $Tucker2$ y $Tucker3$-$HICLAS$, todo modelo (P, Q) $Tucker2$-$HICLAS$ es un modelo (P, Q, R) $Tucker3$-$HICLAS$,con $R \leq min(K, PQ)$. Reciprocamente, cada modelo (P, Q, R) $Tucker3$-$HICLAS$ se puede convertir en un modelo (P, Q) $Tucker2$-$HICLAS$ (Ceulemans and Van Mechelen, 2004).

3.5.1. ALGORITMO

Dado que un modelo (P, Q) $Tucker2$-$HICLAS$ es un modelo (P, Q, R) $Tucker3$-$HICLAS$ con $R \leq$ mín(K, PQ), parece bastante lógico que el modelo (P, Q) $Tucker2$-$HICLAS$ se pueda estimar mediante el algorítmo $Tucker3$-$HICLAS$. Por ejemplo, se puede estimar un modelo (P, Q, R) $Tucker3$-$HICLAS$ con $R =$ mín(K, PQ) y sumar sobre r en la regla de asociación Eq.3.5 del $Tucker3$-$HICLAS$. Aunque por razones de eficiencia (por ejemplo, el hecho de tener que estimar tres matrices en lugar de cuatro), existe un algorítmo específico para el $Tucker2$-$HICLAS$.

Dado una $I \times J \times K$ matriz de datos binarios $\underline{\mathbf{D}}$ y un determinado rango (P, Q), el objetivo del algorítmo $Tucker2$-$HICLAS$ es encontrar una $I \times J \times K$ matriz del modelo $\underline{\mathbf{M}}$ que pueda ser representado mediante un modelo $Tucker2$-$HICLAS$ y que minimice la función de pérdida

$$f(\underline{\mathbf{M}}) = \sum_{i=1}^{I} \sum_{j=1}^{J} \sum_{k=1}^{K} (d_{ijk} - m_{ijk})^2 \tag{3.10}$$

Al igual que el resto de los algorítmos de clases jerárquicas, el algorítmo ejecuta sucesivamente dos rutinas. La primera rutina busca las matrices $\mathbf{A}$, $\mathbf{B}$ y $\underline{\mathbf{G}}$ que combinadas mediante 3.8 hacen que 3.10 sea mínimo. Esta primera rutina comienza a partir de una configuración inicial de las matrices paquete. Esta configuración inicial se puede obtener realizando un análisis $HICLAS$ (De Boeck and Rosenberg (1988)) de las matriciaciones

$I \times JK$ y $J \times IK$ de la matriz de datos $\underline{\mathbf{D}}$ en rangos P y Q respectivamente. La core matrix óptima $\underline{\mathbf{G}}^{(0)}$ que minimiza 3.10 se calcula a partir de la configuración inicial $\mathbf{A}^{(0)}$, $\mathbf{B}^{(0)}$. En el siguiente paso $\mathbf{A}^{(w)}$ se recalcula a partir de $\mathbf{B}^{(w-1)}$ y $\underline{\mathbf{G}}^{(w-1)}$; $\mathbf{B}^{(w)}$ se recalcula a partir de $\mathbf{A}^{(w)}$ y $\underline{\mathbf{G}}^{(w-1)}$; y $\mathbf{G}^{(w)}$ se recalcula a partir de $\mathbf{A}^{(w)}$ y $\underline{\mathbf{B}}^{(w)}$ $(w = 1, 2, ...)$. Se repite este proceso hasta que no se produzcan variaciones significativas en la función de pérdida 3.10.

Para la estimación de las matrices paquete y de la core matrix se puede usar la propiedad de separabilidad de la función de pérdida (Chaturvedi and Carroll (1994)). Esta propiedad dice que la contribución a la función de pérdida del patrón paquete de un objeto i, a_i se puede separar de la contribución de los patrones paquete de los otros objetos. Por tanto, la estimación de $\mathbf{A}$ (condicionada a $\mathbf{B}$ y $\underline{\mathbf{G}}$) se puede obtener estimando consecutivamente los patrones paquete de los I objetos; para lo que se utilizará Regresión Booleana (Leenen and Van Mechelen (1998)). De forma similar se procederá para la estimación de $\mathbf{B}$ y $\underline{\mathbf{G}}$.

Las matrices $\mathbf{A}$, $\mathbf{B}$ y $\underline{\mathbf{G}}$ obtenidas en la primera rutina se han obtenido unicamente con la restricción de que verifiquen la relación de asociación en $\underline{\mathbf{M}}$. Por tanto en la segunda rutina se transforman las matrices $\mathbf{A}$, $\mathbf{B}$ y $\underline{\mathbf{G}}$ para que verifiquen además las relaciones de equivalencia y jerarquía. Para ello se aplicará de forma sucesiva una operación de cierre sobre $\mathbf{A}$, $\mathbf{B}$ y $\underline{\mathbf{G}}$, de manera que se transformarán los ceros de $\mathbf{A}$, $\mathbf{B}$ y $\underline{\mathbf{G}}$ en unos siempre y cuando el cambio no modifique $\underline{\mathbf{M}}$ (ni por tanto la función de pérdida).

3.6. MODELO *TUCKER*1-*HICLAS*

El modelo *Tucker*1-*HICLAS* reduce sólo uno de los tres modos de $\underline{\mathbf{M}}$ (Ceulemans and Van Mechelen, 2005). Por tanto existen tres tipos diferentes de modelos *Tucker*1-*HICLAS*: el tipo A que reduce los objetos, el tipo B que reduce los atributos y el tipo C que reduce los recursos.

Las reglas de descomposición de los modelos Tipo A, B y C *Tucker*1-*HICLAS* son las siguientes:

Tipo A:

$$m_{ijk} = \bigoplus_{p=1}^{P} a_{ip} g_{jkp}$$

Tipo B:

$$m_{ijk} = \bigoplus_{q=1}^{Q} b_{jq} g_{ikq}$$

Tipo C:

$$m_{ijk} = \bigoplus_{r=1}^{R} c_{kr} g_{ijr}$$

siendo P, Q, R el rango de los respectivos modelos.

Se puede comprobar que un modelo *Tucker*1-*HICLAS* de tipo A y rango P con matriz del modelo $\underline{\mathbf{M}}$ es equivalente a un modelo *HICLAS* (De Boeck and Rosenberg, 1988; Van Mechelen et al., 1995) de rango P con matriz del modelo M_a de tamaño $I \times JK$. Analogamente un modelo *Tucker*1-*HICLAS* de Tipo B y rango Q (respectivamente Tipo C y rango R) es equivalente a un modelo *HICLAS* de rango Q (respectivamente R) para M_b (respectivamente M_c).

3.7. RELACIONES ENTRE LOS MODELOS DE CLASES JERÁRQUICAS PARA DATOS BINARIOS EN TRES VÍAS

.

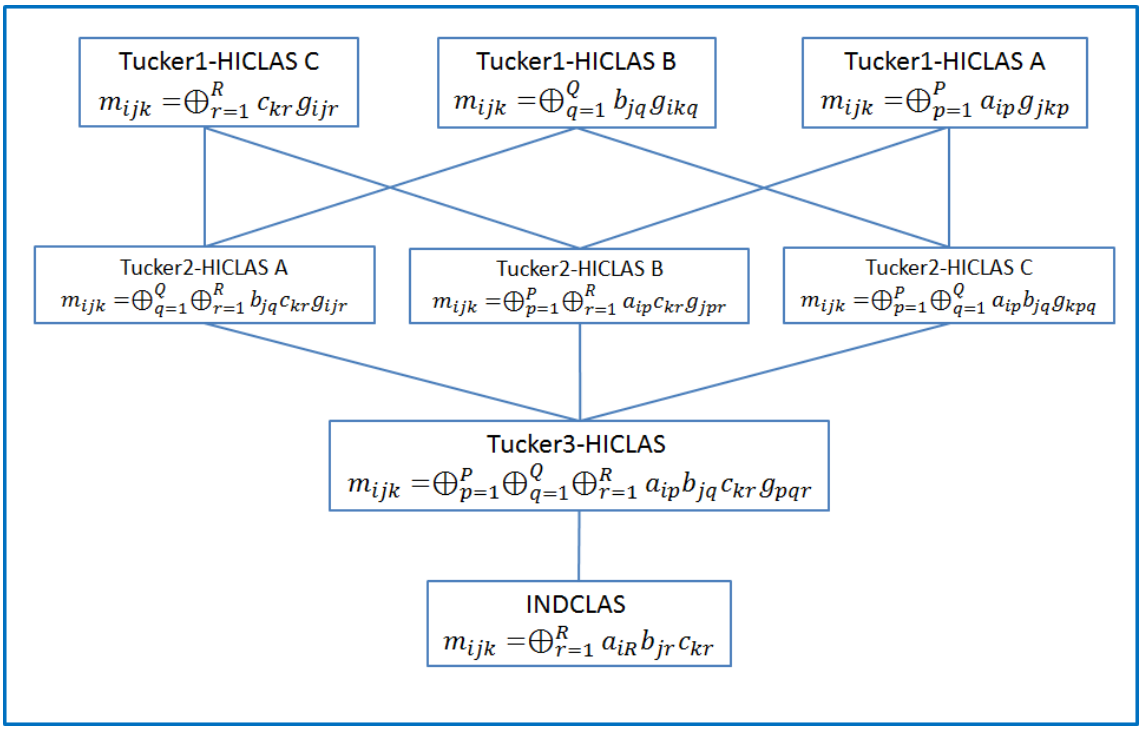


Figura 3.5: Relaciones jerárquicas entre los ocho tipo de modelos de clases jerárquicas para datos binarios en tres vías

En la figura 3.5 se muestran las relaciones entre los distintos tipos de modelos de clases jerárquicas para datos binarios en tres vías.

3.8. APLICACIÓN A DATOS DEL MODELO *TUCKER3-HICLAS*

En esta sección se va a presentar un aplicación práctica del Modelo *Tucker3-HICLAS*. Los datos con los que se va a trabajar proceden de un estudio sobre el estrés en alumnos de las Facultades de Medicina y Farmacia de la Universidad de Salamanca. Se utiliza el Inventario de Estrés Académico de Hernández et al. (1996) (ver ANEXO), el cual pretende identificar, del conjunto de situaciones académicas, aquellas que provocan un mayor nivel de estrés, así como, qué respuestas de estrés se asocian a cada situación, si dichas manifestaciones son las mismas en todas las situaciones o si predomina más algún componente de respuesta (cognitivo, fisiológico o conductual) que otro en función de la situación.

Este cuestionario incluye once situaciones potencialmente generadoras de estrés (e.g., la realización de un examen, la exposición de trabajos en clase, intervención en el aula,...) y para cada una de ellas se presenta una escala de valores de 1 a 5 (donde 1 representa Nada de estrés y 5 Mucho estrés). Por otro lado, asociadas a cada situación, se plantean doce tipos de respuestas de estrés (e.g., inquietud, molestias en el estómago, ganas de llorar,...) que también se evalúan en una escala de 1 a 5, donde 1 significa Casi nunca o nunca y 5 Casi siempre o siempre.

De entre los 387 alumnos que cumplimentan el cuestionario se seleccionan para este estudio los 114 alumnos que han manifestado sentir alguna vez estrés en las once situaciones planteadas. Además se dicotomizan los doce tipos de respuestas de estrés, se asigna valor 0 a Nunca o Rara vez y valor 1 a Algunas veces, Casi siempre ó Siempre.

Más especificamente, en el estudio 114 personas (sources) indican si ó no han sentido cada una de las 12 respuestas de estrés (atributos) en cada una de las 11 situaciones estresantes (objetos). Esto da lugar a una matriz de datos binarios $11 \times 12 \times 114$, $\underline{\mathbf{D}}$, donde $d_{ijk} = 1$ si la persona k manifiesta la respuesta j ante la situación i, y 0 en otro caso (Figura 3.6).

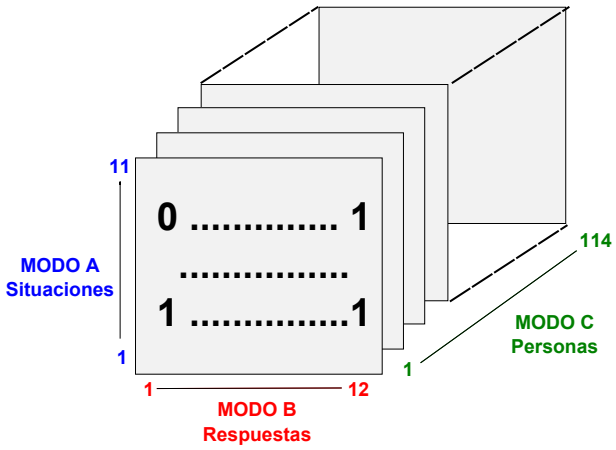


Figura 3.6: Estructura de datos para la aplicación $Tucker3$-$HICLAS$.

Nuestros objetivos van a ser: (a) encontrar clusters de personas (tipos de personas) que reaccionen de forma similar ante las situaciones estresantes; (b) clusters de situaciones ante las cuales los individuos reaccionen de forma similar; (c) clusters de reacciones que ocurren simultaneamente en personas y situaciones. Además el algorítmo proporciona reglas de asociación entre estos tres conjuntos de clusters que permiten determinar si un determinado tipo de persona muestra un cluster de reacciones ante un determinado cluster de situaciones.

Para llevar a cabo dicho análisis se utilizan los programas $Tucker$-$HICLAS$ y $TuckerSpecificSolution$ desarrollados en la Universidad de Leuven (Bélgica).

Se llevan a cabo análisis $Tucker3$-$HICLAS$ desde rango (1,1,1) hasta rango (5,5,5). Las mejores soluciones se muestran en la Tabla 3.8. En dicha Tabla se muestran para cada una de las soluciones el número de discrepancias (positivas y negativas), el número de concordancias 1-1, la proporción de discrepancias y la proporción de discrepancias.

Para facilitar la interpretabilidad de los resultados se discutirá la solución (2,2,3). En términos de ajuste, esta solución representa correctamente el 55,6 % de los datos.

	Rank	Number of discr.	Number of 1-1 conc.	Prop. discr.	Prop. conc.	Scree	Gen. Scree
3	(1,1,1)	4196	3874	0.279	0.257		
5	(1,2,2)	3805	3877	0.253	0.258	0.002	0.008
6	(2,2,2)	3646	4264	0.242	0.283	0.002	0.007
7	(2,2,3)	3515	4396	0.234	0.292	0.002	0.007
8	(3,2,3)	3408	4276	0.226	0.284	0.003	0.006
9	(3,3,3)	3344	4420	0.222	0.294	-0.003	0.005
10	(3,4,3)	3237	4595	0.215	0.305	-0.002	0.006
11	(3,4,4)	3106	4773	0.206	0.317	0.008	0.007
12	(3,4,5)	3097	4963	0.206	0.330	-0.002	0.006
13	(3,5,5)	3065	4891	0.204	0.325	-0.003	0.005
14	(5,4,5)	2987	4993	0.198	0.332	0.005	0.007
15	(5,5,5)	2979	4952	0.198	0.329		

Tabla 3.8: Ajuste de las mejores soluciones del análisis $Tucker3\text{-}HICLAS$

Situaciones estresantes

En Tabla 3.9 se muestran las situaciones estresantes utilizadas y a qué cluster(s) pertenecen cada una de ellas según el análisis $Tucker3\text{-}HICLAS$. En base a esto, se agrupan las situaciones en clases, de manera que las situaciones de cada clase pertenezcan al mismo conjunto de clusters. Existen dos clusters: (1) Cluster 1: donde se agrupan las situaciones de la clase 2 (2) Cluster 2: donde se agrupan las situaciones de las clases 1 y 2.

Las situaciones de la clase 1 (Subir al despacho del profesor en horas de tutorías, Masificación de las aulas, Competitividad entre compañeros y Trabajo en grupo) podríamos calificarlas como 'poco estresantes', mientras que las de la clase 2 (Realización de un examen, Exposición de trabajos en clase, Intervención en el aula, Sobrecarga académica, Falta de tiempo para cumplir con las actividades académicas, Realización de trabajos obligatorios para aprobar la asignatura y Tarea de estudio) podrían calificarse de 'muy estresantes'.

Situaciones	*Cluster 1*	*Cluster 2*
Situaciones Clase 1		
Subir al despacho del profesor en horas de tutorías	0	1
Masificación de las aulas	0	1
Competitividad entre los compañeros	0	1
Trabajo en grupo	0	1
Situaciones Clase 2		
Realización de un examen	1	1
Exposicición de trabajos en clase	1	1
Intervención en el aula	1	1
Sobrecarga académica	1	1
Falta de tiempo para poder cumplir con las actividades académicas	1	1
Realización de trabajos obligatorios para aprobar la asignatura	1	1
Tarea de estudio	1	1

Tabla 3.9: Resumen de las situaciones agrupadas de acuerdo a la clasificación $Tucker3$-$HICLAS$

Reacciones

En la Tabla 3.10 se muestran las reacciones utilizadas y a qué cluster(s) pertenece cada una de ellas según el análisis $Tucker3$-$HICLAS$. En base a esto, se agrupan las reacciones en clases, de manera que las reacciones de cada clase pertenezcan al mismo conjunto de clusters. En este caso existe una clase 0, que según se ve en la Tabla 3.10 no pertenece a ningún cluster. A esta clase pertenecen las reacciones 'Fumo, como o bebo demasiado' y 'Se me seca la boca y tengo dificultades para tragar', reacciones que no presentan los alumnos en las situaciones consideradas. La clase 1 incluye las reacciones 'Siento molestias en el estómago' y 'Siento ganas de llorar', la clase 2 las reacciones 'Realizo movimientos repetitivos con alguna parte de mi cuerpo, me quedo paralizado o mis movimientos son lentos', 'me tiemblan las manos o las piernas', 'me cuesta expresarme verbalmente o a veces tartamudeo'. Finalmente la clase 3 incluye las reacciones 'Inquietud', 'El corazón me late muy rápido y/o me falta aire y la respiración es agitada', 'Siento miedo', 'Siento pensamientos o sentimientos negativos' y 'Me siento inseguro de mi mismo'.

Reacciones	*Cluster 1*	*Cluster 2*
Reacciones Clase 0		
Fumo, como o bebo demasiado	0	0
Se me seca la boca y tengo dificultades para tragar	0	0
Reacciones Clase 1		
Siento molestias en el estómago	1	0
Siento ganas de llorar	1	0
Reacciones Clase 2		
Realizo movimientos repetitivos con alguna parte de mi cuerpo, me quedo paralizado o mis movimientos son torpes	0	1
Me tiemblan las manos o las piernas	0	1
Me cuesta expresarme verbalmente o a veces tartamudeo	0	1
Reacciones Clase 3		
Inquietud	1	1
El corazón me late muy rápido y/o me falta aire y la respiración es agitada	1	1
Siento miedo	1	1
Siento pensamientos o sentimientos negativos	1	1
Me siento inseguro de mi mismo	1	1

Tabla 3.10: Resumen de las reacciones agrupadas de acuerdo a la clasificación *Tucker3-HICLAS*

Estructura de las personas

El modelo discrimina 5 tipos diferentes de personas: Personas tipo 0 (28, 24,56 % de los participantes) no experimentan ninguna de las reacciones consideradas en ninguna de las situaciones; los otros 4 tipos de personas, Personas tipo 1, 2, 3 y 4, contienen 41 (35,97 %), 14 (12,28 %), 13 (11,40 %) y 18 (15,79 %) participantes, respectivamente. Cada uno de estos tipos de personas se caracterizan por una serie de reacciones ante las situaciones estresantes.

En la Figura 3.7 se muestran las reacciones que experimenta cada uno de los tipos de personas ante cada uno de los tipos de situaciones.

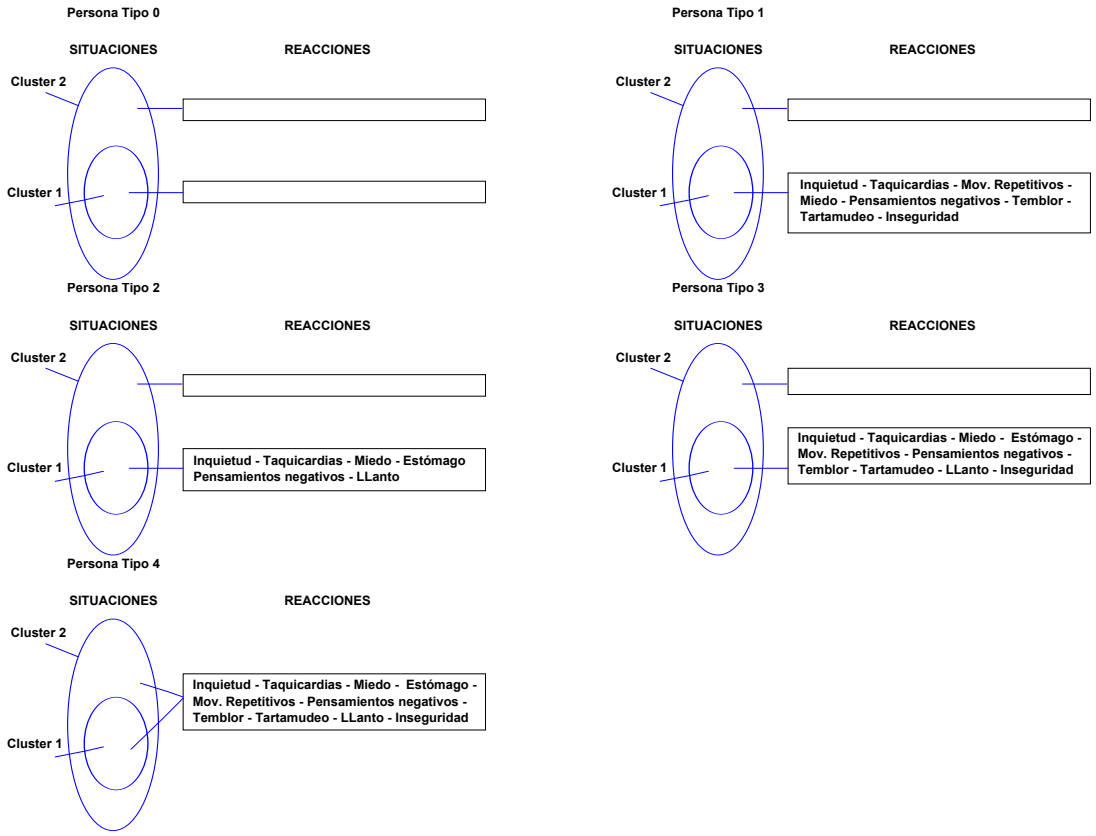


Figura 3.7: Reacciones experimentadas por los distintos tipos de personas en las distintas situaciones. En la parte izquierda de la gráfica aparecen los cluster de situaciones y en la parte derecha las reacciones para cada tipo de situación.

4

ANÁLISIS DE DATOS DE TIPO ORDINAL EN DOS VÍAS: *HICLAS-R*

4.1. INTRODUCCIÓN

Hasta ahora se han utilizado los modelos de la familia de clases jerárquicas con datos binarios. Para poder utilizarla con datos politómicos y en concreto con datos que toman valores enteros de 0 a V, se dicotomizan los datos. Con el fin de evitar la pérdida de información se puede considerar la posibilidad de realizar múltiples dicotimizaciones. Así en una tabla de dos vías, del tipo objetos por atributos, cada atributo se puede reemplazar por una serie de variables. Sin embargo esto produce un efecto colateral ya que uno de los modos se expansiona y puede obstaculizar la interpretación de los resultados (Van Mechelen et al., 2007).

En este capítulo se presenta una extensión de los modelos de clases jerárquicas denominado HICLAS-R para datos de clasificación. Este modelo incluye una generalización de las relaciones de equivalencia, jerarquía y asociación, y además incluye una representación gráfica completa.

Supongamos $\mathbf{D}$ una matriz de datos de dos vías $I \times J$ que toma valores enteros en el rango de 0 a V. El análisis de clases jerárquicas aproxima $\mathbf{D}$ mediante una matriz $\mathbf{M}$ de dimensiones $I \times J$ que toma valores enteros en el rango de 0 a V y que puede ser representada mediante un modelo HICLAS-R. El modelo HICLAS-R tiene que representar las tres relaciones existentes en los modelos HICLAS para datos binarios: (a) equivalencia (b) jerarquía (c) asociación.

El modelo disyuntivo (P, Q, R) HICLAS-R de una $I \times J$ matriz reconstruida $\mathbf{M}$ consta de una $I \times P$ matriz binaria $\mathbf{A}$, una $J \times Q$ matriz binaria $\mathbf{B}$ y una $P \times Q$ matriz core $\mathbf{G}$ con valores en un rango que tiene exactamente $R \leq V$ valores distintos de cero dentro del conjunto de valores $V = \{0, 1, ..., V\}$.

Para ilustrar esta sección se va a utilizar la matriz de datos que se presenta en la Tabla 4.1 (matriz 4×3 con valores en el conjunto $\{0, 1, 2, 3\}$).

Las relaciones de equivalencia, jerarquía y asociación (definidas en el Capítulo 3) se

	Modo 2		
Modo 1	α	β	γ
a	1	0	1
b	0	2	1
c	0	2	1
d	0	3	1

Tabla 4.1: Matriz de datos (Van Mechelen et al., 2007)

representan en las matrices de la siguiente forma: (a) Elementos equivalentes del Modo 1 (2) tienen filas identicas en **A** (**B**). (b) Dos elementos/clases del Modo 1 (2) están jerárquicamente relacionadas si y sólo si sus vectores fila en **A** (**B**) verifican que el vector de la primera es menor o igual que el vector de la segunda. (c) Para la asociación del elemento i-ésimo del Modo 1 con el j-ésimo del Modo 2 se verifica que

$$m_{ij} = Max_{p=1}^{P} Max_{q=1}^{Q} a_{ip} b_{jq} g_{pq} \tag{4.1}$$

La ecuación 4.1 implica que un elemento del Modo 1 está asociado con un elemento del Modo 2 al máximo nivel indicado en la matriz core para un par de paquetes a los que pertenecezcan los elementos.

La Tabla 4.2 contine el modelo disyuntivo HICLAS-R de rango (3, 3, 3) para la matriz de la Tabla 4.1. En concreto la tabla contiene las matrices paquete y la core matrix.

A				**B**				**G**			
a	1	0	0	α	1	0	0	1	0	0
b	0	1	0	β	0	0	1	0	1	2
c	0	1	0	γ	1	1	0	0	0	3
d	0	1	1							

Tabla 4.2: Modelo *HICLAS-R* disyuntivo de rango (3, 3, 3) para los datos de la Tabla 4.1 (Van Mechelen et al., 2007)

En la Tabla 4.2 podemos ver como por ejemplo a los modos b y c que son equivalentes en el Modo 1, le corresponden filas idénticas en **A**. Los modos b y c están jerárquicamente por debajo del modo d en **A** como también ocurría en la Tabla 4.1.

También podemos ver como se reproducen los elementos de la Tabla 4.1:
$m_{42} = Max_{p=1}^{P} Max_{q=1}^{Q} a_{4p} b_{2q} g_{pq} = a_{43} b_{23} g_{33} = 1 * 1 * 3 = 3$

La Figura 4.1 contiene la representación gráfica del modeo *HICLAS-R* de la tabla 4.2.

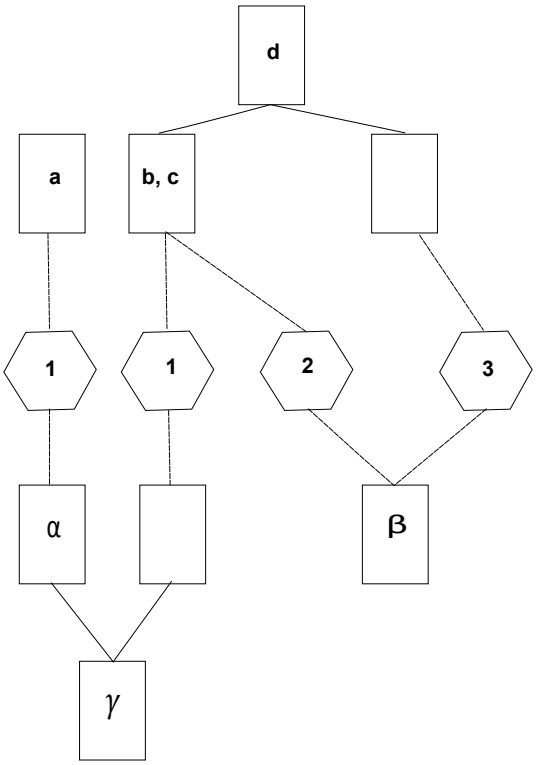


Figura 4.1: Representación Gráfica del Modelo *HICLAS-R* de la Tabla 4.2. Los elementos equivalentes aparecen en cajas. La jerarquía de las clases fila se representa en la mitad superior de la figura, mientras que la jerarquía de las clases columna se representa en la parte inferior (Van Mechelen et al., 2007).

En la Figura 4.1 podemos observar como tenemos tres clases de objetos {a}, {b,c} y {d} (que está jerárquicamente por encima de la clase {b,c}). También existen tres clases de atributos $\{\alpha\}$, $\{\beta\}$ y $\{\gamma\}$. Estas clases se relacionan según se indica en la matriz **G**, de modo que los elementos de primera componente de **A** se relacionan con los de la primera componente de **B** con valor 1: {a} con $\{\alpha, \gamma\}$; los elementos de la segunda componente de **A** se relacionan con los de la segunda componente de **B** con valor 1: {b,c,d} con

$\{\gamma\}$; los elementos de la segunda componente de **A** se relacionan con los de la tercera componente de **B** con valor 2: {b,c,d} con $\{\beta\}$ y finalmente los elementos de la tercera componente de **A** se relacionan con los de la tercera componente de **B** con un 3: {d} con $\{\beta\}$.

4.2. ALGORITMO

El objetivo del análisis disyuntivo *HICLAS-R* en rango (P, Q, R) de una matriz de datos de clasificación es aproximar **D** por medio de una matriz **M** que pueda ser representada mediante un modelo *HICLAS-R* de rango (P, Q, R). Para medir la bondad de la aproximación se utilizará la función de perdida basada en la menor desviación absoluta:

$$L = \sum_{i,j} |m_{ij} - d_{ij}| \tag{4.2}$$

Para ajustar de forma óptima un modelo *HICLAS-R* de rango (P, Q, R) a un conjunto de datos es útil realizar una formulación equivalente del modelo *HICLAS-R*. Esta se obtiene mediante la función t que mapea el conjunto de todas las matrices $I \times J$ con entradas en $V = \{0, 1, ..., V\}$ en el conjunto de todos los arrays binarios $I \times J \times V$.

$$V^{I \times J} \rightarrow \{0,1\}^{I \times J \times V}$$
$$M \rightarrow t(M) \text{ con}$$

$$t(M)_{ijv} = \begin{cases} 1 & \text{si} \quad m_{ij} \geq v \\ 0 & \text{en otro caso} \end{cases}$$

Esto implica que

$$m_{ij} = \sum_{v=1}^{V} t(M)_{ijv} \tag{4.3}$$

En el ejemplo de la Tabla 4.1 la matriz obtenida por la transformación t sería la Tabla 4.3:

Además también se verifica la propiedad siguiente:

$$\forall v' \leq v : t(M)_{ijv} \leq t(M)_{ijv'} \tag{4.4}$$

Modo 1	α	β	γ	α	β	γ	α	β	γ
a	1	0	1	0	0	0	0	0	0
b	0	1	1	0	1	0	0	0	0
c	0	1	1	0	1	0	0	0	0
d	0	1	1	0	1	0	0	1	0

Tabla 4.3: Matriz de datos transformada según t

Además la transformación t preserva la relación de equivalencia en los dos modos, de manera que dos elementos del Modo 1 (2) son equivalentes en M si y solo si los correspondientes elementos del Modo 1 (2) en t(M) son equivalentes. De la misma manera t también preserva las relaciones de jerarquía: un elemento del Modo 1 (2) está jerarquicamente por debajo de un segundo elemento de ese modo si y solo si los correspondientes elementos en t(M) están jerarquicamente relacionados.

Un modelo *HICLAS-R* de rango (P, Q, R) para una matriz reconstruida $I \times J$ de datos de clasificación es por tanto equivalente a un modelo disyuntivo *Tucker*3-*HICLAS* de rango (P, Q, R) del $I \times J \times V$ array binario recodificado t(M). De ahí que tengamos matrices binarias $\mathbf{A}$, $\mathbf{B}$ y $\mathbf{C}$ de tamaños $I \times P$, $J \times Q$ y $V \times R$ respectivamente y una core matriz $\underline{\mathbf{G}}$ de tamaño $P \times Q \times R$.

Las relaciones de equivalencia y jerarquía de t(M) se representan mediante las matrices $\mathbf{A}$, $\mathbf{B}$ y $\mathbf{C}$. En cuanto a la relación de asociación se verifica que

$$t(M)_{ijv} = \bigoplus_{p=1}^{P} \bigoplus_{q=1}^{Q} \bigoplus_{r=1}^{R} a_{ip} b_{jq} c_{vr} \tilde{g}_{pqr} \tag{4.5}$$

y por tanto:

$$m_{ij} = \sum_{v=1}^{V} \bigoplus_{p=1}^{P} \bigoplus_{q=1}^{Q} \bigoplus_{r=1}^{R} a_{ip} b_{jq} c_{vr} \tilde{g}_{pqr} \tag{4.6}$$

Que se puede reescribir como:

$$m_{ij} = \sum_{v=1}^{V} Max_{p=1}^{P} Max_{q=1}^{Q} Max_{r=1}^{R} a_{ip} b_{jq} c_{vr} \tilde{g}_{pqr} \tag{4.7}$$

La ecuación 4.7 se puede reescribir como:

$$m_{ij} = Max_{p=1}^{P} Max_{q=1}^{Q} Max_{r=1}^{R} a_{ip} b_{jq} \tilde{g}_{pqr} \sum_{v=1}^{V} c_{vr} \tag{4.8}$$

$$m_{ij} = Max_{p=1}^{P} Max_{q=1}^{Q} a_{ip} b_{jq} g_{pq} \tag{4.9}$$

donde $g_{pq} = Max_{r=1}^{R} \tilde{g}_{pqr} \sum_{v=1}^{V} c_{vr}$.

El algorítmo que permite ajustar un modelo de rango (P, Q, R) *HICLAS-R* a un conjunto de datos D es el siguiente:

Paso 1 Se hace una recodificación binaria de los datos.

Paso 2 Se ajusta un modelo *Tucker*3-*HICLAS* de rango (P, Q, R) a los datos transformados según $t(\mathbf{D})$, verificando la restricción impuesta en la ecuación 4.4

Paso 3 Convertir la solución final del paso 2 en una formulación del modelo *HICLAS-R* mediante la ecuación 4.5.

El paso clave del algorítmo, el paso 2, se puede descomponer en los siguientes subpasos:

Paso 2a La configuración inicial para las matrices **A**, **B**, y **C** se obtiene mediante el análisis INDCLAS de t(**D**) (Leenen et al., 1999) y a través el análisis disyuntivo *HICLAS* de las tres matriciaciones de t(**D**).

Paso 2b Se recalculan las matrices $\underline{\widetilde{\mathbf{G}}}$, **A**, **B** y **C** haciendo uso del procedimiento de estimación condicional alternativa.

Paso 2c Se transforman las matrices **A** y **B** obtenidas en el paso anterior para que representen correctamente las relaciones de equivalencia y de jerarquía en t(**D**).

La estimación condicionada de $\underline{\widetilde{\mathbf{G}}}$, **A** y **B** del paso 2b se realiza mediante regresión booleana (Leenen and Van Mechelen, 1998); para **A** y **B** la regresión se puede hace para cada fila de forma independiente debido a la propiedad de separabilidad de la función de

pérdida 4.2. Para estimar $\mathbf{C}$ se debe tener en cuenta que ha de satisfacer la constraint 4.4. Esto se consigue buscando el valor óptimo para cada una de las filas de C sujeto a la restricción de que para $\forall r = 1, ..., R : c_{vr} \leq c_{(v-1)r}$.

5

ANÁLISIS DE DATOS ACOPLADOS

En los diferentes campos de investigación es muy común encontrarnos con conjuntos de bloques de datos que tienen uno o más modos en común. En la Figura 5.1 se muestra un ejemplo de dos bloques de datos (una matriz de dos vías y una de tres vías) que comparten un modo en común (genes) (Van Mechelen and Smilde, 2010).

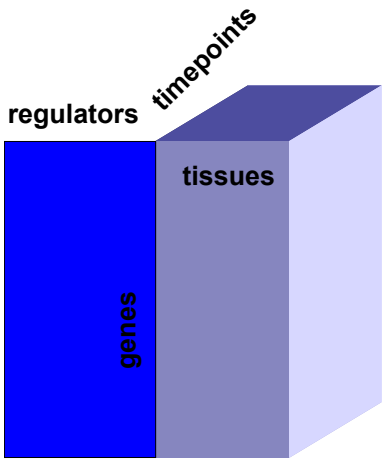


Figura 5.1: Datos acoplados con un modo en común

También puede ocurrir que los bloques de datos compartan múltiples modos como ocurre en la Figura 5.2 ó que compartan parcialmente un modo (Figura 5.3) (Van Mechelen and Smilde, 2010).

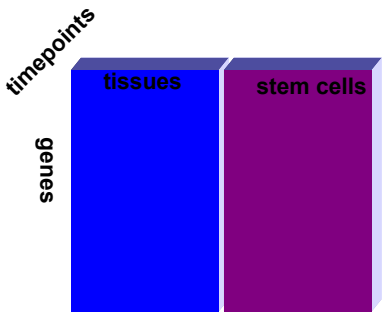


Figura 5.2: Datos acoplados compartiendo más de un modo

.

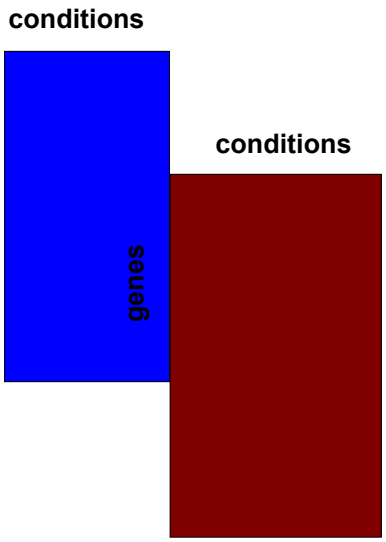


Figura 5.3: Datos de expresión de genes derivados de dos organismos distintos

A la hora de analizar este tipo de datos acoplados es necesario tener en cuenta el rol que juegan los diferentes bloques de datos dentro del análisis global (Van Mechelen and Smilde, 2010). La primera cuestión es si los bloques de datos asumen diferentes roles dentro del análisis global. Si los bloques de datos asumen el mismo rol se dice que son intercambiables. Cuando no son intercambiables hablamos de que existe un bloque de datos predictor y un bloque de datos criterio. Un ejemplo de este tipo de análisis lo encontramos en Omberg et al. (2007). La segunda cuestión es si los diferentes bloques de datos tienen la misma o distinta prioridad.

Existen algunos factores que pueden complicar este análisis: heterogeneidad en tamaño y heterogeneidad en el nivel de ruido entre los diferentes bloques de datos.

5.1. FAMILIA DE MODELOS DE COMPONENTES MULTIVÍA-MULTIBLOQUE

Los objetivos de esta familia de modelos son los siguientes:

- La reconstrucción de la información completa tal y como se incluye en cada bloque de datos.
- Investigación de las similitudes y diferencias entre los distintos bloques de datos y relaciones entre los mismos.
- Los bloques de datos son intercambiables en cuanto a su función y nivel de prioridad (es decir se excluye los modelos de regresión).

Supongamos un conjunto de datos acoplados $\mathbf{D}$, con K bloques de datos relacionados ($\mathbf{B}_1$,...,$\mathbf{B}_k$,...,$\mathbf{B}_K$) (donde el bloque $\mathbf{B}_k$ incluye N_k modos). El modelo global buscado incluye:

1. Un submodelo para cada uno de los bloques de datos.
2. Una estructura que relaciona estos submodelos.

Submodelo para cada uno de los bloques de datos

Supongamos un único bloque de datos $\mathbf{B}$ que constituye una $(I_1 \times ... \times I_n \times ... \times I_N)$ matriz de datos con N vías y N modos. El submodelo que buscamos incluye dos elementos: una cuantificación para cada uno de los modos que incluye el bloque de datos y una regla de asociación que permita reconstruir cada elemento del bloque de datos en base a las cuantificaciones de los modos.

La cuantificacicón de un modo puede entenderse como una reducción de su dimensión. Supongamos que para el n-ésimo modo la cuantificación se recoge mediante la matriz $\mathbf{A}^n$ de tamaño $I_n \times P_n$.

Una vez cuantificados cada uno de los N modos del bloque de datos mediante matrices $\mathbf{A}^n$ de tamaño $I_n \times P_n$ es necesario definir una regla de asociación que incluye un core array $(P_1 \times ... \times P_n \times ... \times P_N)$ $\mathbf{W}$ y una función de mapeo f de manera que:

$$\mathbf{B} = f(\mathbf{A}^1, ..., \mathbf{A}^N, \mathbf{W}) + \mathbf{E} \tag{5.1}$$

donde $\mathbf{E}$ denota un $I_1 \times ... \times I_n \times ... \times I_N$ array con los residuales de error. A continuación se muestran algunos ejemplos de reglas de asociación. El primer ejemplo de regla de asociación es una generalización del producto Cartesiano:

$$\begin{aligned} f(\mathbf{A}^1, ..., \mathbf{A}^N, \mathbf{W})_{i_1 ... i_n ... i_N} = \\ = \sum_{p_1=1}^{P_1} ... \sum_{p_n=1}^{P_n} ... \sum_{p_N=1}^{P_N} a^1_{i_1 p_1} ... a^n_{i_n p_n} ... a^N_{i_N p_N} w_{p_1 ... p_n ... p_N} \end{aligned} \tag{5.2}$$

En el caso de matrices de dos vías dos modos la ecuación 5.2 queda:

$$f(\mathbf{A}^1, \mathbf{A}^2, \mathbf{W})_{i_1 i_2} = \sum_{p_1=1}^{P_1} \sum_{p_2=1}^{P_2} a^1_{i_1 p_1} a^2_{i_2 p_2} w_{p_1 p_2} \tag{5.3}$$

o en forma matricial:

$$f(\mathbf{A}^1, \mathbf{A}^2, \mathbf{W})_{i_1 i_2} = \mathbf{A}^1 \mathbf{W} (\mathbf{A}^2)^T \tag{5.4}$$

En el caso en que las matrices $\mathbf{A}^1$, $\mathbf{A}^2$ tomen valores 0/1 estaríamos hablando de los *biclustering models* (Van Mechelen et al., 2004; Madeira and Oliveira, 2004).

Cuando las matrices de cuantificación toman valores reales la ecuación 5.2 se corresponde con la familia de modelos de datos multivía *Tucker-N*. El caso más común es el correspondiente a tres vías:

$$f(\mathbf{A}^1, \mathbf{A}^2, \mathbf{A}^3, \mathbf{W})_{i_1 i_2 i_3} = \sum_{p_1=1}^{P_1} \sum_{p_2=1}^{P_2} \sum_{p_3=1}^{P_3} a^1_{i_1 p_1} a^2_{i_2 p_2} a^3_{i_3 p_3} w_{p_1 p_2 p_3} \tag{5.5}$$

Una rama de modelos distinta se obtiene cuando la función de mapeo f es una distancia. Por ejemplo para dos vías se puede considerar el modelo

$$f(\mathbf{A}^1, \mathbf{A}^2, \mathbf{W})_{i_1 i_2} = [\sum_{p_1=1}^{P_1} \sum_{p_2=1}^{P_2} (a^1_{i_1 p_1} - a^2_{i_2 p_2})^2 w_{p_1 p_2}]^{\frac{1}{2}} \tag{5.6}$$

Si la matriz W es la matriz identidad, la ecuación 5.5 quedaría

$$f(\mathbf{A}^1, \mathbf{A}^2)_{i_1 i_2} = [\sum_{p=1}^{P} (a^1_{i_1 p} - a^2_{i_2 p})^2)]^{\frac{1}{2}} \tag{5.7}$$

Esta ecuación formaliza el modelo conocido con el nombre de *multidimensional unfolding*.

Estructura que relaciona los distintos submodelos

Supongamos un conjunto de datos que engloba K bloques relacionados que denotamos por $(\mathbf{B}_1, ..., \mathbf{B}_k, ..., \mathbf{B}_K)$, donde el bloque $\mathbf{B}_k$ engloba N_k modos. Para cada uno de los bloques de datos asumimos un submodelo de manera que tenemos:

$$\begin{aligned}
\mathbf{B}_1 &= f_1(\mathbf{A}_1^1, ..., \mathbf{A}_1^{N_1}, \mathbf{W}_1) + \mathbf{E}_1 \\
&... \\
\mathbf{B}_k &= f_k(\mathbf{A}_k^1, ..., \mathbf{A}_k^{N_k}, \mathbf{W}_k) + \mathbf{E}_k \\
&... \\
\mathbf{B}_K &= f_K(\mathbf{A}_K^1, ..., \mathbf{A}_K^{N_K}, \mathbf{W}_K) + \mathbf{E}_K
\end{aligned}$$

donde los subíndices de matrices y funciones se refieren a los bloques y los superíndices de las matrices a los modos dentro de los bloques. En la ecuación anterior las matrices de cuantificación $\mathbf{A}$, la core matrix $\mathbf{W}$ y las funciones f todas ellas llevan subíndice, por lo tanto pueden ser distintas en cada bloque.

El hecho de que los bloques $(\mathbf{B}_1, ..., \mathbf{B}_k, ..., \mathbf{B}_K)$ formen un conjunto de datos acoplados significa que comparten un determinado número de modos. En el modelo global este hecho se refleja introduciendo constraints en las matrices de cuantificación de los modos compartidos. Estas restricciones se pueden ver como la estructura de relación del modelo.

Se pueden considerar una gran variedad de constraints pero la más simple de todas ellas es la identidad. Esta restricción implica simplemente que los modos compartidos se cuantifican de igual forma en todos los submodelos donde aparecen.

Un posible modelo para datos acoplados en tres matrices de dos vías que comparten un modo en común sería el siguiente:

$$\begin{aligned}
\mathbf{B}_1 &= f_1(\mathbf{A}^1, \mathbf{A}_1^2, \mathbf{W}_1) + \mathbf{E}_1 \\
\mathbf{B}_2 &= f_2(\mathbf{A}^1, \mathbf{A}_2^2, \mathbf{W}_2) + \mathbf{E}_2 \\
\mathbf{B}_3 &= f_3(\mathbf{A}^1, \mathbf{A}_3^2, \mathbf{W}_3) + \mathbf{E}_3
\end{aligned}$$

Un posible modelo para datos acoplados donde existen dos bloques, uno de ellos con modos y el otro con tres modos acoplados por uno de ellos sería el siguiente:

$$\begin{aligned} \mathbf{B}_1 &= f_1(\mathbf{A}^1, \mathbf{A}_1^2, \mathbf{W}_1) + \mathbf{E}_1 \\ \mathbf{B}_2 &= f_2(\mathbf{A}^1, \mathbf{A}_2^2, \mathbf{A}_2^3, \mathbf{W}_2) + \mathbf{E}_2 \end{aligned}$$

Algunos ejemplos

Un ejemplo lo tenemos en el método de fusión de datos *SUM-PCA* (*Consensus-PCA*) (Smilde et al., 2003), muy utilizado en quimiometría. Para el caso de dos bloques, una vez realizado el preprocesamiento de datos necesario en cada uno de los bloques, el modelo *SUM-PCA* es el siguiente:

$$\theta_{m1}\mathbf{B}_1 = \mathbf{A}^1(\mathbf{A}_1^2)^T + \mathbf{E}_1 = f(\mathbf{A}^1, \mathbf{A}_1^2) + \mathbf{E}_1$$
$$\theta_{m2}\mathbf{B}_2 = \mathbf{A}^1(\mathbf{A}_2^2)^T + \mathbf{E}_2 = f(\mathbf{A}^1, \mathbf{A}_2^2) + \mathbf{E}_2$$

donde θ_{m1} y θ_{m2} son los pesos que se asignan a los diferentes bloques de datos (Van Deun et al., 2009). Este es un ejemplo de modelo de fusión con link igual a la identidad donde los bloques $\mathbf{B}_1$ y $\mathbf{B}_2$ comparten un modo en común. Este caso se puede generalizar facilmente al caso en que haya más de dos bloques de datos (Heijne et al., 2005). También existen modelos para dos bloques de datos acoplados donde uno de ellos es una matriz de dos vías y el otro una matriz de tres vías que comparten un modo (Wilderjans et al., 2009a). Un ejemplo donde se combinan los modelos *PARAFAC* y *PCA* es el siguiente:

$$\begin{aligned}\mathbf{B}_1 &= \mathbf{A}^1(\mathbf{A}_1^2)^T + \mathbf{E}_1 = f_1(\mathbf{A}^1, \mathbf{A}_1^2) + \mathbf{E}_1\\ \mathbf{B}_2 &= \mathbf{A}^1(\mathbf{A}_2^2 \odot \mathbf{A}_2^3)^T + \mathbf{E}_2 = f_2(\mathbf{A}^1, \mathbf{A}_2^2, \mathbf{A}_2^3) + \mathbf{E}_2\end{aligned}$$

donde $\mathbf{A}_2^2$ y $\mathbf{A}_2^3$ son las matrices correspondientes al segundo y tercer modo de la matriz de tres vías $\mathbf{B}_2$ y $\odot$ es el producto de Katri-Rao.

Finalmente otro ejemplo lo tenemos en el *linked-mode PARAFAC* propuesto por Harshman and Lundy (1994) para bloques de datos acoplados de tres vías. En el caso particular de tres bloques de datos de tres vías donde el primero y el segundo bloque comparten el primer modo y el segundo y el tercero bloque comparten el segundo modo tendríamos:

$$\begin{aligned}\mathbf{B}_1 &= \mathbf{A}^1(\mathbf{A}_1^2 \odot \mathbf{A}_1^3)^T + \mathbf{E}_1 = f(\mathbf{A}^1, \mathbf{A}_1^2, \mathbf{A}_1^3) + \mathbf{E}_1\\ \mathbf{B}_2 &= \mathbf{A}^1(\mathbf{A}^2 \odot \mathbf{A}_2^3)^T + \mathbf{E}_2 = f(\mathbf{A}^1, \mathbf{A}^2, \mathbf{A}_2^3) + \mathbf{E}_2\\ \mathbf{B}_3 &= \mathbf{A}_3^1(\mathbf{A}^2 \odot \mathbf{A}_3^3)^T + \mathbf{E}_3 = f(\mathbf{A}_3^1, \mathbf{A}^2, \mathbf{A}_3^3) + \mathbf{E}_3\end{aligned}$$

5.2. ANÁLISIS DE DATOS ACOPLADOS DE TIPO BINARIO: MODELO *CHIC*

5.2.1. INTRODUCCIÓN

Para analizar datos acoplados podemos seguir dos estrategias (Wilderjans et al., 2005). La primera, que puede ser llamada segmentada, tiene dos pasos consecutivos: primero se ajusta un modelo al bloque de datos que es de interés principal y en segundo lugar se utiliza la cuantificación que se obtiene del modo común en el análisis de los otros bloques de datos. Esta estrategia implica diferentes funciones de pérdida, una para cada submodelo, que se optimizan de forma separada y secuencial. En la segunda estrategia, que puede llamarse integrada, se ajusta un modelo global al conjunto de datos acoplados. Este modelo puede constar de diferentes submodelos, uno para cada bloque de datos. Los modos que son comunes a los diferentes bloques de datos se representan por un conjunto de parámetros comunes, y para su estimación se utiliza la información contenida en todos los bloques de datos. En este caso se optimiza una función de pérdida global, que opcionalmente puede ser la suma de las funciones de pérdida individuales (una para cada submodelo), o también una suma ponderada de las funciones de pérdida individuales donde los pesos están en función de la influencia que se quiera dar a cada bloque de datos en la estimación de los parámetros.

CHIC es un modelo para el análisis integrado de datos acoplados que consisten en una matriz binaria de dos vías y una matriz binaria de tres vías que tienen un modo en común (Figura 5.4). Este modelo contiene dos submodelos, ambos pertenecientes a la familia de modelos de clases jerarquicas para datos binarios de N vías (De Boeck and Rosenberg, 1988; Leenen et al., 1999; Ceulemans et al., 2003). Representa la matriz de datos binarios de tres vías mediante un modelo INDCLAS (Leenen et al., 1999) y el array de datos binario de dos vías mediante un modelo *HICLAS* (De Boeck and Rosenberg, 1988) y relaciona ambos modelos imponiendo la restricción de que los cluster obtenidos para el modo común sean los mismos en ambos submodelos.

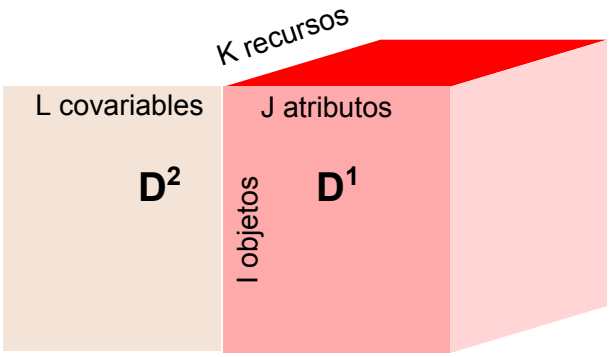


Figura 5.4: Estructura de un conjunto de datos acoplados

5.2.2. MODELO

El modelo combinado *HICLAS-INDCLAS* para datos acoplados (*CHIC*) fue propuesto por Wilderjans et al. (2008). Aproxima un array de datos binarios $\underline{\mathbf{D}}^1$ (de tamaño $I \times J \times K$) y una matriz binaria de datos $\mathbf{D}^2$ (de tamaño $I \times L$) mediante un array binario del modelo $\underline{\mathbf{M}}^1$ (de tamaño $I \times J \times K$) y una matriz binaria del modelo $\mathbf{M}^2$ (de tamaño $I \times L$) de manera que: (1) $\underline{\mathbf{M}}^1$ y $\mathbf{M}^2$ se puedan descomponer mediante un modelo *INDCLAS* y un modelo *HICLAS* respectivamente de rango P (2) la matriz de paquetes objeto $\mathbf{A}$ es la misma en ambos modelos (sin pérdida de generalidad se ha considerado el modo objeto como modo común). El término covariable denota el modo no común de $\mathbf{M}^2$.

Una característica importante del modelo *CHIC* es que incluye una estructura única para el modo común lo que permite relacionar la estructura subyacente en $\mathbf{D}^2$ con la estructura subyacente en $\underline{\mathbf{D}}^1$. Esto implica que los objetos comunes se pueden caracterizar en términos de (1) su asociación con atributos y recursos y (2) su asociación con las covariables.

Al igual que ocurre con los modelos *INDCLAS* y *HICLAS*, el modelo *CHIC*

representa los tres tipos de relaciones estructurales (asociación, equivalencia y jerarquía) en $\underline{\mathbf{M}}^1$ y $\mathbf{M}^2$.

Para ilustrarlas con un ejemplo vamos a usar un modelo hipotético $\underline{\mathbf{M}}^1$ y $\mathbf{M}^2$ que aparece en las Tablas 5.1 y 5.2. El modelo $CHIC$ de rango 3 aparece en la Tabla

Recurso 1	Atributos					Recurso 2	Atributos					Recurso 3	Atributos				
Objetos	A1	A2	A3	A4	A5	Objetos	A1	A2	A3	A4	A5	Objetos	A1	A2	A3	A4	A5
O1	1	1	1	1	1	O1	1	1	0	1	1	O1	1	1	1	1	1
O2	1	1	1	1	1	O2	1	1	0	1	1	O2	1	1	1	1	1
O3	0	0	0	0	0	O3	1	0	0	0	0	O3	1	0	0	0	0
O4	1	1	0	1	1	O4	1	1	0	1	1	O4	1	1	0	1	1
O5	0	0	1	1	0	O5	0	0	0	0	0	O5	0	0	1	1	0
O6	1	1	1	1	1	O6	1	1	0	1	1	O6	1	1	1	1	1
O7	0	0	1	1	0	O7	1	0	0	0	0	O7	1	0	1	1	0
O8	1	1	0	1	1	O8	1	1	0	1	1	O8	1	1	0	1	1

Tabla 5.1: Matriz $\underline{\mathbf{M}}^1$ hipotética (Wilderjans et al., 2005)

Objetos	Covariables							
	C1	C2	C3	C4	C5	C6	C7	C8
O1	1	1	1	1	0	1	1	1
O2	1	1	1	1	0	1	1	1
O3	0	1	0	0	1	0	0	1
O4	0	0	1	1	0	0	1	1
O5	1	1	1	0	0	1	0	1
O6	1	1	1	1	1	1	1	1
O7	1	1	1	0	1	1	1	1
O8	0	1	1	1	1	0	1	1

Tabla 5.2: Matriz $\mathbf{M}^2$ hipotética (Wilderjans et al., 2005)

Asociación El modelo $CHIC$ representa la relación ternaria entre objetos, atributos y recursos, y la relación binaria entre objetos y covariables mediante la reglas de descomposición siguientes:

$$m^1_{ijk} = \bigoplus_{p=1}^{P} a_{ip} b^1_{jp} c_{kp} \tag{5.8}$$

A				**B**^{1}				**C**				**B**^{2}			
	Obj B_1	Obj B_2	Obj B_3		Atr B_1	Atr B_2	Atr B_3		Rec B_1	Rec B_2	Rec B_3		Cov B_1	Cov B_2	Cov B_3
O1	1	1	0	A1	0	1	1	S1	1	1	0	C1	1	0	0
O2	1	1	0	A2	0	1	0	S1	0	1	1	C1	1	0	1
O3	0	0	1	A3	1	0	0	S3	1	1	1	C3	1	1	0
O4	0	1	0	A4	1	1	0					C4	0	1	0
O5	1	0	0	A5	0	1	0					C5	0	0	1
O6	1	1	1									C6	1	0	0
O7	1	0	1									C7	0	1	1
O8	0	1	1									C8	1	1	1

Tabla 5.3: Modelo $CHIC$ de rango 3 para las matrices $\underline{\mathbf{M}}^1$ y $\mathbf{M}^2$ de las tablas 5.1 y 5.2 (Wilderjans et al., 2005)

$$m^2_{il} = \bigoplus_{p=1}^{P} a_{ip} b^2_{lp} \tag{5.9}$$

donde P denota el rango del modelo $CHIC$. Estas reglas de descomposición son equivalentes a:

$$m^1_{ijk} = 1 \Leftrightarrow \exists p : a_{ip} = 1 \wedge b^1_{jp} = 1 \wedge c_{kp} = 1 \tag{5.10}$$

$$m^2_{il} = 1 \Leftrightarrow \exists p : a_{ip} = 1 \wedge b^2_{lp} = 1 \tag{5.11}$$

Esto implica que (1) un objeto i está asociado con un atributo j y un recurso k en $\underline{\mathbf{M}}^1$ ($m^1_{ijk} = 1$) si y sólo si existe al menos un paquete al que pertenezca el objeto i, atributo j y recurso k (2) un objeto i está asociado a una covariable l $\underline{\mathbf{M}}^2$ ($m^2_{il} = 1$) si y sólo si existe al menos un paquete al que pertenezca el objeto i y la covariable l.

Por ejemplo del modelo $CHIC$ en la Tabla 5.3 se puede deducir que el Objeto 8, Atributo 4 y Recurso 1 están asociados en $\underline{\mathbf{M}}^1$ ya que los tres elementos pertenecen al segundo paquete. Además el Objeto 8 y la Covariable 5 están asociados en $\mathbf{M}^2$, porque ambos elementos pertenecen al tercer paquete.

Equivalencia En cada uno de los modos de $\underline{\mathbf{M}}^1$ y $\mathbf{M}^2$ se define una relación de equivalencia. Un objeto i es equivalente a otro objeto i' si y sólo si los dos objetos están asociados con los mismos pares de atributos y de recursos en $\underline{\mathbf{M}}^1$ y con el mismo conjunto de covariables en $\mathbf{M}^2$. La relación de equivalencia entre atributos,

recursos y covariables se define de forma análoga. El modelo $CHIC$ representa la relación de equivalencia entre objetos, atributos, recursos y covariables en términos de paquetes idénticos en $\mathbf{A}$, $\mathbf{B}^1$, $\mathbf{C}$, y $\mathbf{B}^2$ respectivamente.

Por ejemplo, en las tablas 5.1 y 5.2, se puede ver como el Objeto 1 es equivalente al Objeto 2 tanto en $\underline{\mathbf{M}}^1$ como en $\mathbf{M}^2$, de ahí que ambos objetos tengan idéntico patrón paquete en la tabla 5.3.

Jerarquía Se define una relación de jerarquía en cada modo de $\underline{\mathbf{M}}^1$ y $\mathbf{M}^2$. Un objeto i está jerárquicamente por debajo de un objeto i' si y sólo si (1) en $\underline{\mathbf{M}}^1$ el conjunto de pares de atributos y recursos que están asociados con el objeto i es un subconjunto del conjunto de pares de atributos y recursos que están asociados con el objeto i', (2) en $\mathbf{M}^2$ el conjunto de covariables que están asociadas con el objeto i es un subconjunto del conjunto de covariables que están asociadas con el objeto i', y (3) al menos una de las dos relaciones anteriores es extricta. La relación de equivalencia para atributos, recursos y covariables se define de igual forma. El modelo $CHIC$ representa la relación jerárquica entre objetos, atributos, recursos y covariables mediante relaciones entre los patrones paquetes de las matrices $\mathbf{A}$, B^1, $\mathbf{C}$, y B^2 respectivamente.

Por ejemplo, se puede ver en las tablas 5.1 y 5.2 que el Objeto 3 está jerarquicamente por debajo del Objeto 7 en $\underline{\mathbf{M}}^1$ y $\mathbf{M}^2$; de ahí que el patrón paquete del Objeto 3 es un subconjunto del patrón paquete del Objeto 7 en la tabla 5.3.

La representación gráfica de modelo $CHIC$ relaciona la representación gráfica del modelo INDCLAS de $\underline{\mathbf{M}}^1$ que aparece en la parte inferior del gráfico con la representación gráfica del modelo $HICLAS$ de $\mathbf{M}^2$ que aparece en la parte superior del gráfico mediante la representación de los paquetes del modo común. Por tanto se consigue una doble caracterización de los objetos comunes en términos de (1) pares de atributos-recursos (2) covariables.

La clasificación jerárquica de covariables y atributos aparece en la parte superior e inferior de la representación respectivamente. Las clases de elementos equivalentes se incluyen en cajas en las que se encierran las etiquetas de los elementos que pertenecen a las mismas. Las relaciones jerárquicas entre los elementos (clases) de un modo se

representan mediante líneas que conectan las correspondientes cajas. Ambas clasificaciones se relacionan mediante la representación de los paquetes de objetos (representados mediante rectángulos) y los paquetes de recursos (representados mediante hexágonos). Cada rectángulo (hexágono) además contiene las etiquetas de los objetos (recursos) que pertenecen al correspondiente paquete objeto (recurso). En la Figura 5.5 tenemos la representación gráfica del Modelo *CHIC* de la Tabla 5.3.

La doble caracterización de los objetos en términos de pares de atributos-recursos y covariables se puede ver a partir de la Figura de la forma siguiente: Un objeto i se puede caracterizar en términos de (1) un par atributo-recurso (j,k) y (2) una covariable l sí y sólo si hay un camino consistente en líneas rectas y zigzags, que conectan la covariable l con el atributo j y tal que contiene el objeto i en un rectángulo y el recurso k en un hexágono. Por ejemplo, en la Figura se puede ver que el Objeto 2, se puede caracterizar en terminos de (1) el par atributo-recurso (5-3), y (2) la covariable 7, dado que existe un camino entre la covariable 7 y el atributo 5 que contiene al objeto 2 y el recurso 3.

.

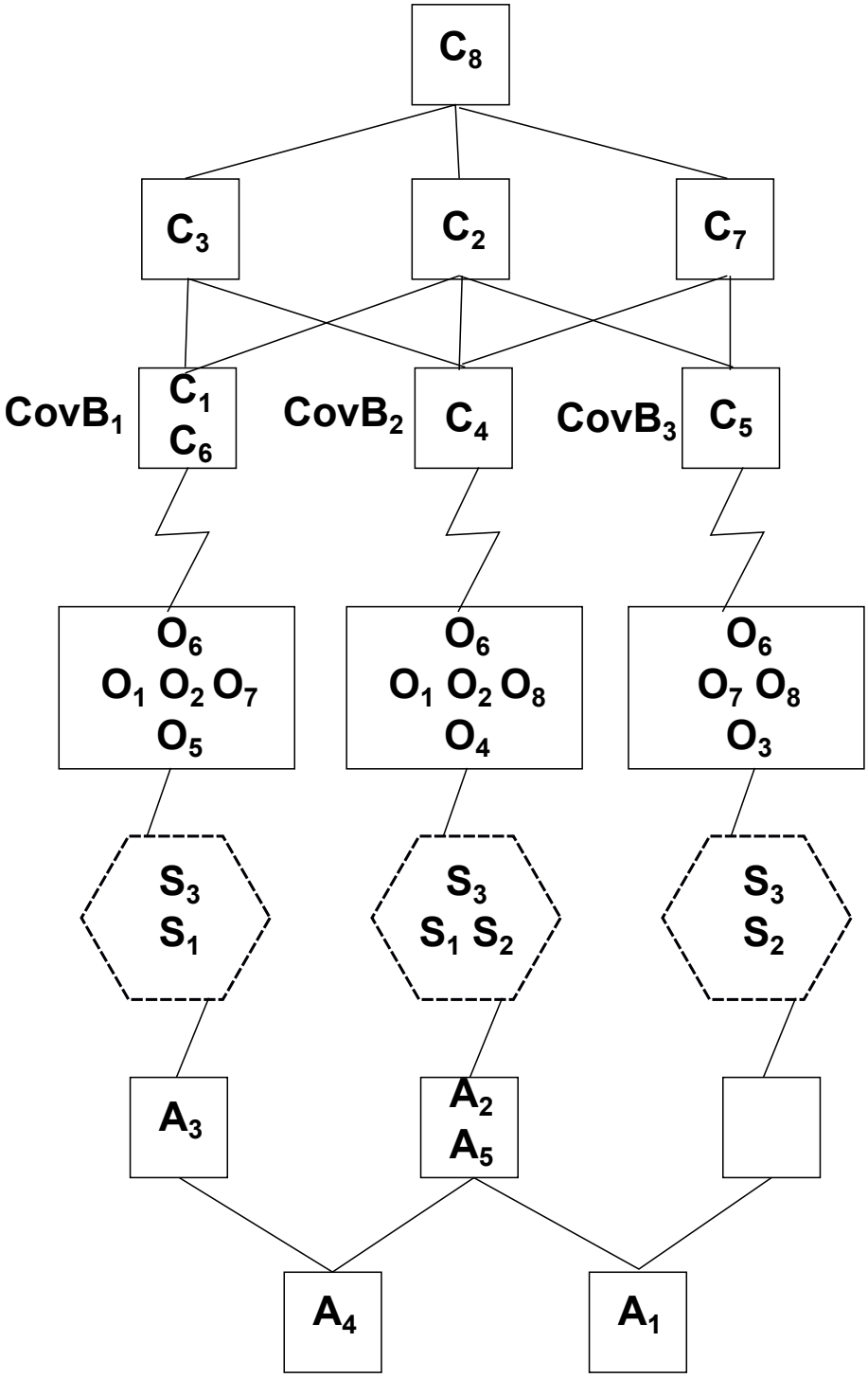


Figura 5.5: Representación gráfica del Modelo *CHIC* de la tabla 5.3 (Wilderjans et al., 2005)

5.2.3. ALGORITMO

El objetivo del análisis $CHIC$ en rango P de un $I \times J \times K$ array de datos binario $\underline{\mathbf{D}}^1$ y una $I \times L$ matriz de datos binaria $\mathbf{D}^2$ es estimar un $I \times J \times K$ array binario del modelo $\underline{\mathbf{M}}^1$ y una $I \times L$ matriz binaria del modelo $\mathbf{M}^2$ de manera que (1) el valor de la función de pérdida

$$f(\underline{\mathbf{M}}^1, \mathbf{M}^2) = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}(d^1_{ijk} - m^1_{ijk})^2 + \sum_{i=1}^{I}\sum_{l=1}^{L}(d^2_{il} - m^2_{il})^2 \tag{5.12}$$

sea mínimo y (2) $\underline{\mathbf{M}}^1$ y $\mathbf{M}^2$ se puedan representar por un modelo INDCLAS de rango P y un modelo $HICLAS$ de rango P, respectivamente, con una matriz paquete en común $\mathbf{A}$. El algorítmo tiene dos fases consecutivas. En la primera fase, se estiman las matrices paquete $\mathbf{A}$, $\mathbf{B}^1$, $\mathbf{C}$ y $\mathbf{B}^2$ de manera que minimicen la función de pérdida 5.12. Al final de la primera fase se restringen las matrices paquete para que representen sólo la relación de asociación en ($\underline{\mathbf{M}}^1$, $\mathbf{M}^2$). En la segunda fase, se ajustan las matrices paquete para que representen las relaciones de equivalencia y jerarquía correctamente, sin alterar la función de pérdida.

5.3. ANÁLISIS *PARAFAC-PCA*

El análisis *PARAFAC-PCA* se usa para analizar una tabla de tres vías con valores reales y una tabla de dos vías con valores reales que tienen un modo en común.

5.3.1. MODELO

El modelo *PARAFAC-PCA* (Wilderjans et al., 2009b) es un modelo para el análisis de un array de datos reales $(I \times J \times K)$ $\underline{\mathbf{D}}^1$ y una matriz de datos $(I \times L)$ $\mathbf{D}^2$ que tienen el primer modo en común. Se aproxima $\underline{\mathbf{D}}^1$ mediante una matriz del modelo *PARAFAC* $\underline{\mathbf{M}}^1$ $(I \times J \times K)$ con P componentes y $\mathbf{D}^2$ se aproxima mediante el modelo *PCA* (Pearson, 1901; Hotelling, 1933) $\mathbf{M}^2$ con P componentes.

Es importante destacar que se restringe la matriz de componentes del primer modo para que sea la misma en ambos modelos. Por tanto podemos escribir $\underline{\mathbf{M}}^1$ y $\mathbf{M}^2$ como sigue:

$$m^1_{ijk} = \sum_{p=1}^{P} a_{ip} b^1_{jp} c^1_{kp}$$

$$m^2_{il} = \sum_{p=1}^{P} a_{ip} b^2_{lp}$$

donde a_{ip}, b^1_{jp}, c^1_{kp} y b^2_{lp} son los elementos de las matrices de componentes $\mathbf{A}$ $(I \times P)$, $\mathbf{B}^1$ $(J \times P)$, $\mathbf{C}^1$ $(K \times P)$ y $\mathbf{B}^2$ $(L \times P)$.

La familia de modelos *multiway multiblock components* (al que pertenece el modelo *PARAFAC-PCA*) está relacionada con la familia de modelos *multiway covariates regression* (Smilde and Kiers, 1999). Estas familias de modelos son muy similares, la diferencia radica en el rol que asigna cada modelo a los bloques de datos. En el modelo *multiway covariates regression* los bloques de datos tienen diferentes roles (es decir,

un bloque sirve como bloque predictor y el otro como bloque criterio), mientras que en el modelo *multiway multiblock components* los bloques de datos son intercambiables conceptualmente. Los modelos *multiway covariates regression* a partir de Smilde et al. (2000) comienzan a llamarse modelos *multiway multiblock regression*.

5.3.2. ANÁLISIS DE DATOS

El objetivo del análisis *PARAFAC-PCA* con P componentes, de datos $(\underline{\mathbf{D}}^1, \mathbf{D}^2)$ es estimar un modelo $(\underline{\mathbf{M}}^1, \mathbf{M}^2)$ de manera que el valor de la función de pérdida:

$$f = \alpha \times \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}(d^1_{ijk} - \sum_{p=1}^{P} a_{ip}b^1_{jp}c^1_{kp})^2 + (1-\alpha) \times \sum_{i=1}^{I}\sum_{l=1}^{L}(d^2_{il} - \sum_{p=1}^{P} a_{ip}b^2_{lp})^2$$

$(0 \leq \alpha \leq 1)$, sea mínimo y $(\underline{\mathbf{M}}^1, \mathbf{M}^2)$ se puedan representar mediante los modelos *PARAFAC* y *PCA* con P componentes, respectivamente.

Los valores α y $1-\alpha$ denotan el peso con el que los datos de las matrices de tres y dos vías influyen en el análisis. Si $\alpha = 1$ entonces las componentes del modo común se determinan sólo en base a la matriz de tres vías, mientras que si $\alpha = 0$ sólo se considera la matriz de dos vías para la estimación de las componentes del modo común.

Para estimar las matrices de componentes del modelo *PARAFAC-PCA* se utiliza un algorítmo basado en mínimos cuadrados alternados. En este algorítmo se parte de una estimación inicial de las matrices de componentes y se recalculan alternativamente cada una de ellas condicionado al valor de las otras hasta que se alcanza un criterio de parada.

La estimación inicial de las matrices de componentes podemos hacerla de dos formas:

1. La estimación inicial de $\mathbf{A}$, $\mathbf{B}^1$ y $\mathbf{C}^1$ se realiza tomando los P vectores propios correspondientes a los P mayores valores propios de $[\mathbf{D}^1_a|\mathbf{D}^2]$, $\mathbf{D}^1_b$ y $\mathbf{D}^1_c$ respectivamente. Donde $\mathbf{D}^1_a$, $\mathbf{D}^1_b$ y $\mathbf{D}^1_c$ son las matriciaciones $I \times JK$, $J \times KI$ y $K \times IJ$ de

$\underline{\mathbf{D}}^1$ y | denota la matriz concatenación.

2. Aleatoriamente

Una vez obtenidas las estimaciones iniciales de $\mathbf{A}$, $\mathbf{B}^1$ y $\mathbf{C}^1$ se obtiene una estimación inicial de $\mathbf{B}^2$ por regresión de $\mathbf{A}$ sobre $\mathbf{D}^2$.

Una vez que se han obtenido las estimaciones iniciales de $\mathbf{A}$, $\mathbf{B}^1$, $\mathbf{C}^1$ y $\mathbf{B}^2$ el algorítmo realiza una serie de iteraciones hasta que alcanza el criterio de parada del algorítmo. En cada iteración, cada matriz de componentes se reestima condicionadamente a las otras matrices de componentes. Así los valores de $\mathbf{A}$ se recalculan a partir de los valores de $\mathbf{B}^1$, $\mathbf{C}^1$ y $\mathbf{B}^2$ que permanecen fijos.

En cuanto al criterio de parada, en el procedimiento de mínimos cuadrados alternados, después de cada iteración se comprueba si el decremento en la función de pérdida es mayor o igual que un valor prefijado denominado tolerancia. Si es así entonces se procede a realizar una nueva iteración. En caso contrario el algorítmo para, se ha alcanzado el criterio de convergencia, y se mantienen las matrices de componentes actuales. Como en el análisis $PARAFAC$ el número de iteraciones hasta converger puede ser muy grande, se suele establecer un número máximo de iteraciones, de manera que el algorítmo para al llegar a ese número de iteraciones a pesar de no haberse alcanzado el criterio de convergencia.

6

MODELO $T3$-PCA

6.1. INTRODUCCIÓN

En el capítulo anterior se ha introducido el concepto de Datos Acoplados y las distintas estrategias que se pueden utilizar para su análisis: (1) estrategia segmentada y (2) estrategia integrada.

Un ejemplo de datos acoplados sería el siguiente: podemos recoger información acerca de cómo reaccionan una serie de personas en determinadas situaciones y al mismo tiempo información sobre los rasgos de personalidad de dichos individuos; los rasgos de personalidad y cómo reaccionan ante ciertas situaciones se pueden considerar como dos piezas de información acerca de la personalidad de un individuo.

En este ejemplo tiene sentido utilizar una estrategia integrada/segmentada ya que los dos conjuntos de datos contienen información relevante para la clasificación de los individuos. Además, cabe esperar que las distintas formas de reaccionar ante ciertas situaciones estén relacionadas con los distintos rasgos de personalidad.

Es de esperar que la estrategia integrada sea mejor que la segmentada especialmente cuando los dos bloques de datos además de la información común contengan una gran cantidad de información distintiva. En este caso la estrategia segmentada posiblemente no recuperará de forma adecuada la información común mientras que si lo hará la estrategia integrada.

Dentro de las últimas (estrategias integradas), hemos visto el Modelo $PARAFAC$-PCA (Wilderjans et al., 2009a) para dos bloques de datos acoplados donde uno de ellos es una matriz de dos vías y el otro una matriz de tres vías que tienen un modo en común. En este modelo se representa el bloque de datos de tres vías mediante un modelo $PARAFAC$ y la matriz de dos vías mediante un modelo PCA, con la restricción de que la matriz de componentes para el modo común ha de ser la misma en ambos submodelos. Como ya hemos visto el modelo $PARAFAC$ es bastante restrictivo (i.e., mismo número de componentes para cada modo y relación uno a uno entre las componentes de los diferentes modos).

En este capítulo se presenta un nuevo modelo, al que se denomina $T3$-PCA, para el Análisis Integrado de Datos Acoplados como alternativa al Modelo $PARAFAC$-PCA. Permitirá por tanto el Análisis de una matriz de datos de tres vías y una matriz de datos de dos vías que tienen un modo en común. Más especificamente consta de dos submodelos, un modelo $Tucker3$ para la matriz de tres vías y un modelo PCA para la matriz de dos vías, y dichos modelos se relacionan imponiendo la restricción de que la matriz de componentes para el modo común ha de ser igual en ambos submodelos. Este modelo es menos restrictivo que el $PARAFAC$-PCA dado que el número de componentes puede ser diferente para cada uno de los modos y se permiten interacciones entre todas la componentes.

El capítulo se organiza según las siguientes secciones: (1) Modelo; (2) Análisis de datos; (3) Selección del modelo; (4) Rotaciones; (5) Estudio de simulación.

En el Capítulo **ARTÍCULOS**, se recoge el artículo sometido a la revista **BEHAVIOR RESEARCH METHODS**, en el que se desarrollan parte de los contenidos de este capítulo.

Data fusion by $T3$-PCA: An integrated global model for the simultaneous analysis of coupled real-valued data

Se incluye aqui un breve resumen del trabajo.

ABSTRACT

In various areas of science, researchers try to get insight into important processes by jointly analysing different data sets containing information regarding common, and possibly also distinctive, aspects of these processes. For example, to explain individual differences in personality, researchers collect for the same set of persons data regarding behavioural signatures (i.e., the reaction profile of a person across different situations), on the one hand, and traits or dispositions, on the other hand. To uncover the processes underlying such coupled data (i.e., two data blocks that have the person mode in common), to all N-way N-mode data blocks simultaneously a global model is fitted in which each data block is represented by a N-way N-mode decomposition model (e.g., PCA, $Parafac$, $Tucker3$) and the parameters underlying the common mode are restricted to be the same for all data blocks this mode belongs to. To estimate the parameters underlying the common mode, an integrated strategy is used that pools the information present in all data blocks (i.e., data fusion). Belonging to the family of component models, one such global model is the recently proposed *Linked-mode Parafac-PCA* ($LMPCA$) model (Wilderjans et al., 2009a,b). For most empirical data, however, $LMPCA$ assumes a too simplistic underlying structure as the three-way data are modelled with the very restrictive $Parafac$ model (i.e., same number of components for each mode and a simple one-to-one correspondence between components of different modes). To overcome these limitations, in this paper, we propose the $T3$-PCA model which represent the three-

and two-way data with $Tucker3$ and PCA, respectively. This model is less restrictive than $LMPCA$ because the number of components is allowed to vary across modes and components of different modes may interact in a more complicated way. To estimate the $T3$-PCA model parameters, an alternating least squares algorithm is proposed. The performance of the integrated $T3$-PCA strategy is evaluated and compared with the performance of a segmented strategy (i.e., using only information from one data block to estimate the parameters of the common mode) in an extensive simulation study.

Keywords data fusion, coupled/linked data, integrated strategy, multiblock multiway data analysis, $Tucker3$, PCA, common and distinctive components, segmented strategy, alternating least squares (ALS).

6.2. MODELO

6.2.1. ESTRUCTURA DE DATOS

El análisis $T3$-PCA requiere de una $(I \times J \times K)$ matriz de datos reales $\underline{\mathbf{D}}^1$ y una $(I \times L)$ matriz de datos también reales $\mathbf{D}^2$ que tienen el primer modo en común. Por ejemplo, en estudios de personalidad, como se puede ver en la Figura 6.1, el array de datos $\underline{\mathbf{D}}^1$ contiene el perfil de comportamiento de un conjunto de participantes en diferentes situaciones, mientras que $\mathbf{D}^2$ contiene las puntuaciones del mismo conjunto de participantes en una lista de items de personalidad. Otros ejemplos de este tipo de datos podemos encontrarlos en psicometría (Wilderjans et al., 2009b,a), neuropsicología, bioinformática (Acar et al., 2012) y quimiometría (Smilde et al., 2000). Para una revisión de los mismos ver Acar et al. (2013). Es importante señalar que el conjunto de elementos del modo común en $\underline{\mathbf{D}}^1$ y en $\mathbf{D}^2$ es el mismo (i.e., el mismo conjunto de participantes en el ejemplo de psicología).

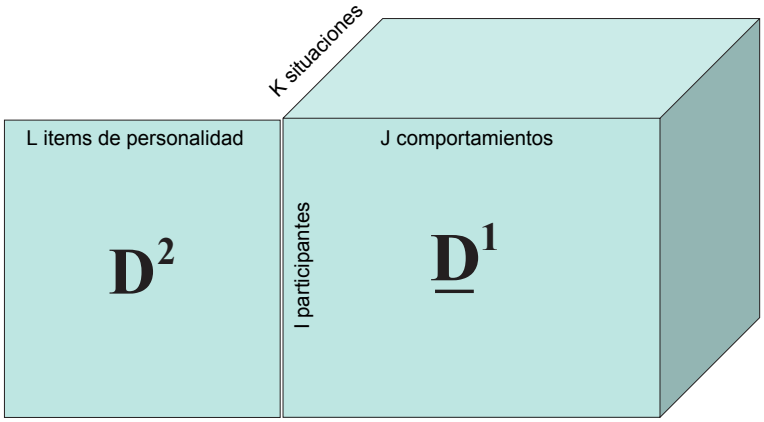


Figura 6.1: Representación gráfica de los datos para un análisis $T3$-PCA.

6.2.2. MODELO GLOBAL PARA EL ANÁLISIS DE DATOS ACOPLADOS DE TIPO REAL: $T3$-PCA

Modelo. El modelo integrado $T3$-PCA aproxima un $(I \times J \times K)$ array de datos de tipo real $\underline{\mathbf{D}}^1$ que está acoplado a través del primer modo con una $(I \times L)$ matriz de datos también reales $\mathbf{D}^2$, mediante una matriz $\underline{\mathbf{M}}^1$ (de tamaño $I \times J \times K$) de datos reales y una matriz $\mathbf{M}^2$ (de tamaño $I \times L$), respectivamente, tales que tienen el primer modo en común. En concreto, $\underline{\mathbf{D}}^1$ se aproxima mediante un $I \times J \times K$ array $\underline{\mathbf{M}}^1$ que se puede descomponer de acuerdo a un modelo Tucker (Tucker, 1966; Kroonenberg and de Leeuw, 1980) con P componentes de objetos, Q componentes de atributos y R componentes de recursos; $\mathbf{D}^2$ se aproxima mediante una matriz $\mathbf{M}^2$ que se puede descomponer de acuerdo a un modelo PCA (Pearson, 1901; Hotelling, 1933) con P componentes, de manera que la matriz de componentes para el primer modo (i.e., las filas) en $\underline{\mathbf{M}}^1$ y $\mathbf{M}^2$ sea idéntica. Por tanto $\underline{\mathbf{M}}^1$ y $\mathbf{M}^2$ se pueden descomponer como sigue:

$$m^1_{ijk} = \sum_{p=1}^{P}\sum_{q=1}^{Q}\sum_{r=1}^{R} g_{pqr} a_{ip} b^1_{jq} c_{kr} \tag{6.1}$$

$$m^2_{il} = \sum_{p=1}^{P} a_{ip} b^2_{lp} \tag{6.2}$$

donde a_{ip}, b^1_{jq}, c_{kr} y b^2_{lp} son los elementos de las matrices de componentes $\mathbf{A}(I \times P)$, $\mathbf{B}^1(J \times Q)$, $\mathbf{C}(K \times R)$ y $\mathbf{B}^2(L \times P)$, y P, Q y R denotan el número de componentes para objetos, atributos y recursos, respectivamente. Los elementos g_{pqr} son los elementos de la core matrix $\underline{\mathbf{G}}$ (de tamaño $P \times Q \times R$) y reflejan la relación entre la p^{th}, q^{th} y r^{th} componente de $\mathbf{A}$, $\mathbf{B}$ y $\mathbf{C}$, respectivamente. Señalar que $\underline{\mathbf{M}}^1$ y $\mathbf{M}^2$ estan relacionadas mediante la restricción de que matriz de componentes de objetos $\mathbf{A}$ sea igual en ambos submodelos.

Es importante señalar que los parámetros del modelo $T3$-PCA (i.e., las matrices de componentes y la core matrix) son únicos salvo permutación, escalado, reflexión y rotación (Smilde et al., 2004). Para identificar parcialmente la solución, restringimos las matrices de componentes $\mathbf{A}$, $\mathbf{B}_1$ y $\mathbf{C}$ de modo que sean ortonormales (i.e., $\mathbf{A}'\mathbf{A} = (\mathbf{B}_1)'\mathbf{B}_1$

$= \mathbf{C}'\mathbf{C} = \boldsymbol{I}$, lo cual implica que las componentes son ortogonales y tienen longitud uno).

Relacion con otros modelos. El modelo $T3$-PCA pertenece a la familia de modelos *multiway multiblock component models*, los cuales incluyen diferentes modelos, como entre otros, el modelo *linked-mode* $PARAFAC$-PCA (Wilderjans et al., 2009a,b).

El nuevo modelo $T3$-PCA está relacionado con muchos otros modelos dentro y fuera de la familia de modelos *multiway multiblock component models*. Dentro de esta familia, el modelo $T3$-PCA se puede considerar una generalización del modelo $PARAFAC$-PCA, de la misma forma que el modelo $Tucker3$ es una generalización del modelo $Parafac$. En concreto, cuando se restringe $P = Q = R$ y $\underline{\mathbf{G}}$ es la matriz superidentidad (i.e., $g_{rst} = 1$ si $r = s = t$ y $g_{rst} = 0$ en otro caso), el modelo $Tucker3 - PCA$ es equivalente al modelo $PARAFAC$-PCA.

La familia de modelos *multilevel multiblock component models* está muy relacionada con la familia de modelos *multiway covariate/multiblock regression models* (Smilde and Kiers, 1999; Smilde et al., 2000). Ambas familias de modelos sólo difieren en el rol que asignan a los diferentes bloques de datos. En el modelo *multiway covariate/multiblock regression models* los diferentes bloques de datos tienen distintos roles (i.e., un bloque sirve como bloque predictor mientras que el otro sirve como bloque criterio), mientras que el modelo *multiway multiblock component models* los dos bloques de datos son intercambiables en términos conceptuales.

Pre-Procesamiento. Previo al análisis, los datos se deben preprocesar mediante el escalado y/o centrado de los mismos. En lo relativo al centrado, en el array de tres vías, la media de cada combinación atributo-recurso (a través de los objetos) debe ser cero, mientras que en la matriz de dos vías cada covariable debe ser centrada (a través de los objetos). Además, para tener en cuenta diferencias de escala en las variables, en el array de tres vías la varianza de cada atributo (a través de objetos y recursos) debe ser uno, mientras que en la matriz de dos vías la varianza de cada covariable debe ser uno.

6.3. ANÁLISIS DE DATOS

6.3.1. OBJETIVO

El objetivo del análisis $T3$-PCA de rango (P, Q, R), de datos $(\underline{\mathbf{D}}^1, \mathbf{D}^2)$ es estimar un modelo $(\underline{\mathbf{M}}^1, \mathbf{M}^2)$ de manera que el valor de la función de pérdida:

$$f = \frac{\alpha}{\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}(d_{ijk}^1)^2} \times \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}(d_{ijk}^1 - \sum_{p=1}^{P}\sum_{q=1}^{Q}\sum_{r=1}^{R} a_{ip}b_{jq}^1 c_{kr} g_{pqr})^2 \qquad (6.3)$$

$$+\frac{(1-\alpha)}{\sum_{i=1}^{I}\sum_{l=1}^{L}(d_{il}^2)^2} \times \sum_{i=1}^{I}\sum_{l=1}^{L}(d_{il}^2 - \sum_{p=1}^{P} a_{ip}b_{lp}^2)^2, (0 \leq \alpha \leq 1)$$

sea mínimo (con $0 \leq \alpha \leq 1$), $\underline{\mathbf{M}}^1$ se pueda representar mediante un modelo $Tucker3$ con P, Q, R componentes ($\mathbf{A}$, $\mathbf{B}_1$ y $\mathbf{C}$ ortonormales), $\mathbf{M}^2$ por un modelo PCA con P componentes y la matriz de componentes $\mathbf{A}$ sea igual en la descomposición de $\underline{\mathbf{D}}^1$ y $\mathbf{D}^2$. Los pesos α y $1-\alpha$ denotan el peso con el que los datos de las matrices de tres y dos vías influyen en el análisis (Wilderjans et al., 2009a). Si $\alpha = 1$ entonces las componentes del modo común se determinan sólo en base a la matriz de tres vías, mientras que si $\alpha = 0$ sólo se considera la matriz de dos vías para la estimación de las componentes del modo común. Para valores de alfa intermedios, las componentes comunes se estiman usando la información (ponderada) de ambos bloques de datos, lo que se entiende como un enfoque integrado.

Con el fin de evitar que la escala de los datos de cada bloque influya en el análisis, dividimos la suma de las diferencias al cuadrado entre los datos y el modelo para cada bloque, por la suma de cuadrados de los datos en cada bloque. Como, en general, el bloque de datos de tres vías tendrá un valor mayor en la suma de los cuadrados de sus entradas, esta normalización implica que la influencia del bloque de tres vías se ve

disminuida.

La función de pérdida anterior (6.3) se puede reescribir como:

$$f_2 = \beta \times \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}(d^1_{ijk} - \sum_{p=1}^{P}\sum_{q=1}^{Q}\sum_{r=1}^{R} a_{ip}b^1_{jq}c_{kr}g_{pqr})^2 + (1-\beta) \times \sum_{i=1}^{I}\sum_{l=1}^{L}(d^2_{il} - \sum_{p=1}^{P} a_{ip}b^2_{lp})^2 \tag{6.4}$$

con $\beta = \frac{\alpha \times \sum_{i=1}^{I}\sum_{l=1}^{L}(d^2_{il})^2}{\alpha \times \sum_{i=1}^{I}\sum_{l=1}^{L}(d^2_{il})^2 + (1-\alpha) \times \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}(d^1_{ijk})^2}$ (Vervloet et al., 2013).

6.3.2. ALGORITMO

Para estimar los parámetros del modelo $T3\text{-}PCA$ se utiliza un algorítmo basado en los mínimos cuadrados alternados (Kroonenberg and de Leeuw, 1980). En este algorítmo, después de obtenerse estimaciones iniciales de todos los parámetros del modelo, se recalculan alternativamente las matrices de componentes $\mathbf{A}$, $\mathbf{B}_1$, $\mathbf{C}$ y $\mathbf{B}_2$ y la core matrix $\underline{\mathbf{G}}$ hasta que se alcanza un criterio de parada (Ten Berge, 1993).

Las estimaciones iniciales de las matrices de componentes se pueden realizar racionalmente o aleatoriamente (para una discusión de los diferentes tipos de inicialización, consultar Ceulemans et al. (2007)). La estimación racional de $\mathbf{A}$, $\mathbf{B}_1$, y $\mathbf{C}$ se obtiene tomando los vectores propios asociados con los P, Q y R valores propios mayores de $[\mathbf{D}_a^1|\mathbf{D}^2][\mathbf{D}_a^1|\mathbf{D}^2]'$, $\mathbf{D}_b^1(\mathbf{D}_b^1)'$ y $\mathbf{D}_c^1(\mathbf{D}_c^1)'$, respectivamente, donde $\mathbf{D}_a^1$, $\mathbf{D}_b^1$ y $\mathbf{D}_c^1$ son las $I \times JK$, $J \times KI$, y $K \times IJ$ matriciaciones de $\underline{\mathbf{D}}^1$ y $|$ denota la concatenación de matrices Kiers (2000). Resaltar que con este procedimiento ya se asegura que los valores iniciales de $\mathbf{A}$, $\mathbf{B}_1$, y $\mathbf{C}$ son ortonormales.

La estimación aleatoria de las matrices de componentes se obtiene, primero, generando entradas de una distribución normal estandar y después, para conseguir matrices de componentes ortonormales, se calcula una base ortonormal del rango de cada una de las matrices de componentes. Una vez que tenemos los valores iniciales de $\mathbf{A}$, $\mathbf{B}_1$ y $\mathbf{C}$, se obtienen las estimaciones iniciales de $\mathbf{B}_2$ y $\underline{\mathbf{G}}$ por medio de regresión lineal.

A partir de las estimaciones iniciales de los parámetros del modelo, el algoritmo realiza una serie de iteraciones, hasta que alcanza el criterio de parada. En cada iteración, se re-estiman las matriz de componentes y la core matrix a partir de los valores de los otros parámetros. En concreto, en primer lugar, se fijan los valores de $\mathbf{B}_1$, $\mathbf{C}$ y $\mathbf{B}_2$, y se reestiman las entradas de $\mathbf{A}$. A continuación, se actualizan las entradas de $\mathbf{B}_1$ (condicionado a $\mathbf{A}$ y $\mathbf{C}$) y $\mathbf{C}$ (condicionado a $\mathbf{A}$ y $\mathbf{B}_1$). Finalmente, se recalculan $\underline{\mathbf{G}}$ (condicionado a $\mathbf{A}$, $\mathbf{B}_1$ y $\mathbf{C}$) y $\mathbf{B}_2$ (condicionado a $\mathbf{A}$). Por tanto la actualización de las matrices de componentes $\mathbf{B}_1$, $\mathbf{C}$, $\mathbf{B}_2$ y la core matrix $\underline{\mathbf{G}}$ se reduce a un problema de regresión lineal multivariante (Kroonenberg and de Leeuw, 1980; Smilde et al., 2004;

Andersson and Bro, 1998; Bro and Andersson, 1998).

Finalmente, $\mathbf{A}$, que se obtiene a partir de un modelo $Tucker3$ para el array de tres vías y el modelo PCA para la matriz de dos vías, se puede reestimar como $\mathbf{A} = \mathbf{U}\mathbf{V}^{'}$ donde $\mathbf{U}$ y $\mathbf{V}$ son los P vectores singulares izquierdos y derechos, respectivamente, asociados con los P mayores valores singulares de $\mathbf{Y}\mathbf{Z}^{'}$, donde $\mathbf{Y} = [\sqrt{\alpha} \times \mathbf{D}_a^1 \mid \sqrt{(1-\alpha)} \times \mathbf{D}^2]$ y $\mathbf{Z} = [\sqrt{\alpha} \times \mathbf{G}_a \, (\mathbf{C} \otimes \mathbf{B}_1)^{'} \mid \sqrt{1-\alpha} \times \mathbf{B}_2^{'}]$.

Después de cada iteración, se recalcula el valor de la función de pérdida, usando los parámetros recién actualizados. Cuando la actualización de las matrices de componentes y la core matrix implica un decremento significativo en la función de pérdida (i.e. un decremento mayor o igual que un valor de tolerancia predeterminado) se procede a realizar una nueva iteración, en otro caso el algoritmo para y se retienen las estimaciones actuales de los parámetros del modelo como soluciones finales.

Dado que la actualización de los parámetros del modelo dentro del proceso iterativo nunca puede llevarnos a un incremento del valor de la función de pérdida y que el valor de la función de pérdida está acotado inferiormente por el cero, está garantizado que el algoritmo $T3$-PCA converge. Esto, sin embargo, no implica que el algoritmo siempre identifique la estimación óptima de los parámetros, ya que podemos acabar en soluciones óptimas locales. Para minimizar el riesgo de que el algoritmo $T3$-PCA acabe en un mínimo local, llevamos a cabo un procedimiento multistart. En este procedimiento, el algoritmo se ejecuta varias veces, cada vez con diferentes valores iniciales de los parámetros obtenidos de forma aleatoria o seudo-aleatoria [1], quedándonos con la solución para la cual el valor de la función de pérdida sea menor.

[1] Los valores iniciales seudo-aleatorios se pueden obtener perturbando ligeramente una solución inicial racional, para más información consultar Ceulemans et al. (2007); Wilderjans and Ceulemans (2013).

6.4. NÚMERO DE COMPONENTES A RETENER

En general, no se dispone de información sobre el rango óptimo (P,Q,R) subyacente a un modelo $T3$-PCA para un determinado conjunto de datos. Para dar solución a este problema, los investigadores pueden realizar varios análisis con diferentes complejidades variando el rango del modelo desde (1,1,1) hasta (P,Q,R) y seleccionar el modelo para el cual exista un buen balance entre el ajuste del modelo y el nivel de complejidad. Podemos cuantificar el desajuste del modelo a través de la función de pérdida, mientras que para medir la complejidad del modelo podemos utilizar diferentes opciones: el número total de componentes (i.e., $P+Q+R$), el número total de parámetros (i.e., $IP+JQ+KR+PQR+LP$) y el número de parámetros libres (i.e., $IP+JQ+KR+PQR+LP-P^2-Q^2-R^2$).

Para determinar el número de componentes a retener para un modelo $T3$-PCA vamos a utilizar el método **CHull** (Ceulemans and Kiers, 2006; Wilderjans et al., 2013). Para aplicar este método necesitaremos ajustar varios modelos $T3$-PCA. Recordar que sólo será necesario ajustar modelos en los que se cumpla que $PQ \geq R$ y $PR \geq Q$ y $QR \geq P$. También se pueden omitir modelos en los que $P > max(I, JK)$, $Q > max(J, IK)$ y/o $R > max(K, IJ)$ (Timmerman and Kiers, 2000).

Para cada uno de estos modelos calcularemos el valor de la función de pérdida (misfit) f y la complejidad c. Como medida de la complejidad en este caso se ha optado por utilizar el número total de componentes $P + Q + R$. Una vez obtenidos f y c para cada uno de los modelos considerados, el método consta de 7 etapas:

1. Para cada nivel de complejidad se retiene el mejor modelo (es decir aquel que tiene menor valor en la función de pérdida). Si hubiera varios con el mismo valor se selecciona uno de ellos aleatoriamente.

2. Se ordenan los modelos retenidos en el paso anterior $m_i (i = 1, ..., n)$, según su complejidad c_i. De manera que el más simple se corresponde con $i = 1$ y el más complejo $i = n$.

3. Se consideran todos los pares de modelos adyacentes, y se excluyen aquellos modelos m_i para los cuales $f_j \leq f_i$ con $j < i$. Se repite este paso hasta que no se puedan excluir más modelos.

4. Para cada triplete de modelos adyacentes (m_i, m_j, m_k) se excluyen los modelos m_j para los cuales $f_j \geq f_i + (c_j - c_i)\frac{f_k - f_i}{c_k - c_i}$, o lo que es lo mismo se excluyen los modelos m_j que están localizados en o por encima de la línea que conecta los otros dos modelos (i.e., m_i y m_k). Se repite este paso hasta que no se eliminen más modelos. Los modelos resultantes estarán todos situados en la parte inferior del convex hull.

5. Para cada uno de los modelos anteriores se calcula el valor *st*:

$$st_i = \frac{\frac{f_i - f_{i-1}}{c_i - c_{i-1}}}{\frac{f_{i+1} - f_i}{c_{i+1} - c_i}}$$

6. Se retiene el modelo para el cual el valor *st* es máximo. Si hubiera dos modelos para los cuales el valor *st* fuera máximo, se retiene el más simple (i.e., el de menor valor c_i).

6.5. POSTPROCESAMIENTO: ROTACIONES

El modelo $Tucker3$, según vimos en capítulos anteriores, puede ser escrito en notación matricial como:

$$\mathbf{X} = \mathbf{AG}(\mathbf{C}' \otimes \mathbf{B}') + \mathbf{E} \tag{6.5}$$

donde $\mathbf{X}$, $\mathbf{G}$ y $\mathbf{E}$ se corresponden con las matrices de tres vías $\underline{\mathbf{X}}$, $\underline{\mathbf{G}}$ y $\underline{\mathbf{E}}$, escritas como matrices de dos vías de orden $(I \times JK)$, $(P \times QR)$ y $(I \times JK)$, respectivamente, y $\otimes$ representa el producto de Kronecker.

La descomposición $Tucker3$ no es única. Sea $\mathbf{S}$ $(P \times P)$ una matriz no singular. Entonces

$$\mathbf{X} = \mathbf{ASS}^{-1}\mathbf{G}(\mathbf{C}' \otimes \mathbf{B}') + \mathbf{E} = \widetilde{\mathbf{A}}\widetilde{\mathbf{G}}(\mathbf{C}' \otimes \mathbf{B}') + \mathbf{E}$$

En otras palabras, podemos modificar la core matrix $\underline{\mathbf{G}}$ sin que se afecte al ajuste simplemente aplicando la modificación inversa a la matriz correspondiente. Esta libertad rotacional nos permitirá elegir transformaciones que simplifiquen la estructura de la core de manera que la mayoría de los elementos de $\underline{\mathbf{G}}$ sean cero, y así de este modo eliminar interacciones entre las correspondientes componentes (Kolda and Bader, 2009).

Dado que tanto el modelo $Tucker3$ como el modelo PCA tienen libertad rotacional, el modelo combinado $T3$-PCA también tiene libertad rotacional. Se pueden rotar las matrices de componentes $\mathbf{A}$, $\mathbf{B}_1$ y $\mathbf{C}$ siempre y cuando la core matrix $\underline{\mathbf{G}}$ (y, en el caso de $\mathbf{A}$, también $\mathbf{B}_2$) sea contra-rotada. Además se pueden rotar cada una de las matrices $\mathbf{A}$ y $\mathbf{B}_2$ siempre que la otra matriz y la core matrix sean adecuadamente contra-rotadas.

La libertad rotacional se puede utilizar por lo tanto para obtener una solucion más facilmente interpretable. Para conseguirlo podemos utilizar diferentes estrategias: (1) Rotar las matrices de componentes (por ejemplo, aplicando una rotación VARIMAX (Kaiser, 1958)) y después rotar adecuadamente la core matrix (2) Rotar la core matrix y después contra-rotar las matrices de componentes (Kiers, 1992, 1997). La primera opción permite una fácil interpretación de las matrices de componentes pero probablemente

la core matrix no será facilmente interpretable, es decir, existirán relaciones complejas entre las componentes. La segunda opción dará lugar justo a lo contrario (fácil interpretación de la core matrix y por tanto de las relaciones entre las componentes pero difícil interpretación de las matrices de componentes).

6.6. ESTUDIO DE SIMULACIÓN

6.6.1. PROBLEMA

En esta sección se presentan las simulaciones que se han realizado con el objetivo de evaluar el rendimiento del algorítmo $T3$-PCA.

Estamos interesados en dos aspectos relativos al rendimiento: optimización y recuperación de datos. En cuanto a la optimización, examinaremos la sensibilidad del algoritmo ante los mínimos locales. Mientras que para la recuperación determinaremos en qué grado el algoritmo es capaz de recuperar los datos y evaluaremos en qué medida el algoritmo $T3$-PCA que utiliza una estrategia integrada (i.e., usa la información del array de tres vías y de la matriz de dos vías para obtener las componentes del modo común) es mejor que una estrategia separada (i.e., obtener las componentes de $\mathbf{A}$ a partir del array de tres vías y después descomponer la matriz de dos vías con la restricción de que la matriz $\mathbf{A}$ ha de ser igual).

Para hacer frente a este último objetivo, es importante distinguir entre las componentes del modo común que explican una cantidad considerable de la varianza en ambos bloques de datos (es decir, las componentes comunes) y las componentes que sólo son subyacentes a uno de los dos bloques de datos (i.e., componentes distintivas) (Van Deun et al., 2013; Schouteden et al., 2013, 2014).

Cuando en el conjunto de datos acoplados sólo subyacen componentes comunes, es de esperar que la estrategia segmentada (ie., aplicar un $Tucker3$ a los datos de tres vías) funcione bastante bien, y practicamente sea imposible que la estrategia integrada tenga un mejor desempeño. La razón de esto es que la estructura $Tucker3$ impuesta a los datos de tres vías es bastante restrictiva, lo que resulta en una estimación adecuada de la matriz de componentes comunes (i.e., el algoritmo $Tucker3$ filtra el ruido en los datos bastante bien, (Wilderjans et al., 2009b, 2008)). Sin embargo, cuando el modo común del bloque de datos de tres vías contiene una componente distintiva fuerte (en términos

de cantidad de variabilidad explicada) y una o más componentes comunes pequeñas, una estrategia segmentada puede tener la tentación de retener la componente distintiva a expensas de las componentes comunes pequeñas debido a que la información del bloque de dos vías no se utiliza y las estrategias de selección de modelos están dirigidas a la selección de las componentes fuertes principalmente (Ceulemans and Kiers, 2009). Como en una estrategia integrada se toma la información, tanto del bloque de datos de tres vías como del de dos vías, podemos predecir que esta estrategia será capaz de ignorar la componente fuerte pero distintiva y retendrá las componente comunes (menos fuertes).

Para poner a prueba nuestra predicción, vamos a establecer un estudio de simulación en el que los dos bloques de datos contienen una componente extra (además de las comunes) que es distintiva y vamos a investigar cómo el rendimiento de recuperación se ve afectado por la fuerza de este elemento distintivo. Además, también vamos a estudiar cómo el rendimiento de la recuperación depende de estas otras cuatro características de los datos:

1. El tamaño del bloque de datos de tres vías;
2. La ratio entre el tamaño del bloque de tres vías y el tamaño del bloque de dos vías;
3. El rango del modelo;
4. La cantidad de ruido de los datos.

Con lo que respecta a estas cinco características esperamos que el rendimiento del algoritmo sea menor cuando aumenta la cantidad de ruido (van den Berg et al., 2009; Wilderjans et al., 2012, 2013) y cuando la componente distintiva sea más importante. Además, esperamos que la estrategia integrada supere a la estrategia segmentada cuando la componente distintiva sea más importante. Por último, también vamos a explorar cómo el rendimiento en la recuperación tiene que ver con la elección de α.

6.6.2. DISEÑO Y PROCEDIMIENTO

Se manipulan los cinco factores introducidos anteriormente en un diseño completamente randomizado:

1. El tamaño $(I \times J \times K)$ del array de tres vías $\underline{\mathbf{T}}^1$, con dos niveles: $(20 \times 20 \times 20)$ y $(20 \times 20 \times 100)$;

2. La ratio r entre el tamaño del array de tres vías $\underline{\mathbf{T}}^1$ y el tamaño de la matriz de dos vías $\mathbf{T}^2$:
$$r = \frac{I \times J \times K}{I \times L} \tag{6.6}$$
Este factor se manipula con los valores 1 y 0,25. Por tanto se consideran cuatro tamaños posibles para el bloque 2D: $(20, 400)$, $(20, 2000)$, $(20, 100)$ y $(20, 500)$;

3. El rango del modelo $(P \times Q \times R)$, con dos niveles $(2 \times 2 \times 2)$, $(2 \times 3 \times 4)$;

4. La cantidad de ruido de los datos, ε, a 2 niveles: 0,15 y 0,30;

5. La cantidad de varianza explicada por una componente extra distintiva (diferente para el bloque de tres vías y el bloque de dos vías). Este factor se manipula a tres niveles: 5 % (i.e., componente distintiva débil), 15 % (i.e., intermedia) y 30 % (i.e., fuerte). En la Tabla 6.1, en la que se muestra la cantidad de varianza explicada por cada componente, se puede observar que la ratio de la varianza explicada por las otras dos componentes (comunes) se mantiene constante e igual a $\frac{70\,\%}{30\,\%}$.

	Común		Distintiva
Nivel	Componente 1	Componente 2	Componente 3
Débil	0,665	0,285	0,050
Intermedia	0,595	0,255	0,150
Fuerte	0,490	0,210	0,300

Tabla 6.1: Proporción de varianza explicada por las componentes subyacentes al modo común para cada uno de los niveles de cantidad de varianza explicada por la componente distintiva.

Para cada una de las posibles combinaciones de los niveles de los cinco factores manipulados, se construyen las matrices de componentes $\mathbf{A}^{(T)}$, $\mathbf{B}_1^{(T)}$, $\mathbf{C}^{(T)}$, $\mathbf{B}_2^{(T)}$ generando valores de la distribución normal estandar; en un segundo paso, estas matrices se centran (i.e., media cero) y se ortonormalizan por columnas (i.e., columnas ortogonales de longitud uno). Para que las componentes subyacentes a la matriz de dos vías cumplan los requerimientos sobre la varianza explicada (ver Tabla 6.1), se escalan las columnas de $\mathbf{B}_2^{(T)}$.

Se generan los valores de $\underline{\mathbf{G}}^{(T)}$ a partir de la distribución normal estandar y, después, se reescalan de manera que la suma de los cuadrados de las filas de $\underline{\mathbf{G}}^{(T)}$ estén en concordancia con los porcentajes de varianza explicada por cada una de las componentes del modo común (ver Tabla 6.1)[2].

Para añadir la componente distintiva a cada bloque de datos, aumentamos $\mathbf{A}^{(T)}$ con una columna diferente (dando lugar a $\mathbf{A}_{3D}^{(T)}$ y $\mathbf{A}_{2D}^{(T)}$, las cuales sólo difieren en la última columna) de manera que cada nueva $\mathbf{A}^{(T)}$ se centra y ortonormaliza por columnas, la correlación entre la componente distintiva de $\mathbf{A}_{3D}^{(T)}$ y $\mathbf{A}_{2D}^{(T)}$ sea cero y la cantidad de varianza explicada por las componentes sea la requerida (ver Tabla 6.1).

A continuación, se construyen los pares de bloques de datos ($\underline{\mathbf{T}}^1$, $\mathbf{T}^2$) aplicando los algoritmos $Tucker3$ y PCA a las matrices anteriores $\mathbf{A}^{(T)}$, $\mathbf{B}_1^{(T)}$, $\mathbf{C}^{(T)}$, $\mathbf{B}_2^{(T)}$ y $\underline{\mathbf{G}}^{(T)}$. Además, para cada par de bloques ($\underline{\mathbf{T}}^1$, $\mathbf{T}^2$), se construye otro par de bloques de datos ($\underline{\mathbf{D}}^1$, $\mathbf{D}^2$) añadiendo las matrices de error $\underline{\mathbf{E}}^1$ ($I \times J \times K$) y $\mathbf{E}^2$ ($I \times L$) a $\underline{\mathbf{T}}^1$ y $\mathbf{T}^2$, respectivamente. $\underline{\mathbf{E}}^1$ y $\mathbf{E}^2$ se generan aleatoriamente, generando valores de una distribución normal estándar y posteriormente se reescalan para conseguir obtener la cantidad de ruido requerida ε.

Este procedimiento de generación de datos se repite 20 veces para obtener 20 replicaciones por celda.

[2]En el modelo $Tucker3$, cuando las matrices de componentes $\mathbf{A}$, $\mathbf{B}$ y $\mathbf{C}$ son ortonormales, los cuadrados de los elementos de la core matrix $\underline{\mathbf{G}}$ son proporcionales a la cantidad de variabilidad explicada por cada combinación de componentes. Por tanto, la suma de los cuadrados de los elementos fila de $\underline{\mathbf{G}}$ están relacionados con la cantidad de varianza de cada componente de $\mathbf{A}$.

Teniendo en cuenta todos los niveles de cada uno de los factores del diseño: 2 (tamaños del bloque 3D) x 2 (ratios entre el tamaño del bloque 2D y el bloque 3D) x 2 (rangos) x 3 (tamaños de la componente extra adicional) x 2 (cantidades de ruido) x 20 (replicaciones) = 960 distintos pares de bloques de datos obtenidos ($\underline{\mathbf{T}}^1$, $\mathbf{T}^2$) y ($\underline{\mathbf{D}}^1$, $\mathbf{D}^2$).

Se aplica el algoritmo $T3$-PCA a cada pareja de bloques de datos ($\underline{\mathbf{D}}^1$, $\mathbf{D}^2$), utilizando el rango (P, Q, R) real y repitiendo el análisis con diez valores diferentes para α: 0,050; $0,100$; $0,300$; $0,500$; $0,600$; $0,700$; $0,800$; $0,900$; $0,950$ y $0,999$.

Para poder comparar la estrategia integrada y la segmentada, se analizan cada pareja de bloques acoplados ($\underline{\mathbf{D}}^1$, $\mathbf{D}^2$) siguiendo el siguiente procedimiento: se aplica un análisis $Tucker3$ a $\underline{\mathbf{D}}^1$ y a continuación un PCA a $\mathbf{D}^2$ con la restricción de que $\mathbf{A}$ ha de ser la misma que la obtenida en el análisis $Tucker3$.

Para aumentar la probabilidad de que tanto el algoritmo $Tucker3 - PCA$ como el algoritmo $Tucker3$ no caigan en un mínimo local se lleva a cabo un procedimiento multistart con una inicialización racional y 50 aleatorias.

El estudio de simulación se lleva a cabo en R (versión 3.0.2).

6.6.3. RESULTADOS

OPTIMIZACIÓN DEL RENDIMIENTO: SENSIBILIDAD A MÍNIMOS LOCALES

El objetivo de esta sección es investigar si el algoritmo $T3$-PCA es capaz de identificar el mínimo global de la función de pérdida (Eq. (6.3)).

Por el hecho de haber añadido ruido a los datos, la solución óptima no es conocida (Wilderjans et al., 2008). Una solución a este problema consiste en la definición de una aproximación al óptimo global (proxy), que puede ser concebida como nuestra mejor estimación de la solución óptima global, y evaluar si el algoritmo $T3$-PCA obtiene una solución cuyo valor para la función de pérdida sea menor o igual que el valor de la función de pérdida para el proxy.

Definimos el proxy como la solución con el menor valor de la función del pérdida de las siguientes soluciones:

1. La solución real (i.e., $\mathbf{A}^{(T)}$, $\mathbf{B}_1^{(T)}$, $\mathbf{C}^{(T)}$, $\underline{\mathbf{G}}^{(T)}$ y $\mathbf{B}_2^{(T)}$)
2. La solución obtenida ejecutando el algoritmo segmentado

Se considera la solución $T3$-PCA como sub-óptima cuando el valor de su función de pérdida es mayor que el proxy. Esto ocurre sólo en 27 de 9.600 conjuntos de datos (0,28 % de los casos). Además para todos estos casos el valor de alfa es 0,999; para alfa igual a 0,5 parece no haber ningún problema con los mínimos locales.

RECUPERACIÓN DE LOS PARÁMETROS DEL MODELO

Esta sección tiene dos objetivos: (1) el estudio del rendimiento en la recuperación del algoritmo $T3$-PCA y (2) investigar en qué medida la estrategia integrada $T3$-PCA mejora a la estrategia segmentada. El rendimiento en la recuperación se estudia a nivel de las matrices de componentes y de la core matrix.

Para medir el rendimiento en la recuperación de las matrices de componentes se utiliza el estadístico goodness-of-recovery statistic (GOR) para cada una de las matrices de componentes. En concreto para la matriz de componentes común $\mathbf{A}$, se define el GOR como:

$$GOR_{\mathbf{A}} = \frac{\sum_{p=1}^{P} |\phi(a_p^{(T)}, a_p^{(M)})|}{P}$$

donde ϕ es el coeficiente phi de Tucker (Tucker, 1951), $a_p^{(T)}$ y $a_p^{(M)}$ son la componente p-ésima de la matriz de componentes generada $\mathbf{A}^T$ y de la matriz de componentes $\mathbf{A}^M$ resultante del análisis $T3$-PCA, respectivamente, y P es el número de componentes. Para tener en cuenta la libertad rotacional de las matrices de componentes (Sección 6.5), se encuentra la rotación ortogonal de la matriz de componentes $\mathbf{A}^M$ que mejor ajuste a la matriz $\mathbf{A}^T$ en el sentido de los mínimos cuadrados alternados. La definición del estadístico GOR es similar para las demás matrices de componentes $\mathbf{B}_1$, $\mathbf{C}$, y $\mathbf{B}_2$. El valor del estadístico GOR toma valores entre 0 y 1, correspondiendo el valor 1 a una recuperación perfecta.

El rendimiento en la recuperación a nivel de la core matrix se determina mediante la media de las diferencias al cuadrado entre la core matrix generada $\underline{\mathbf{G}}^{(T)}$ y la core matrix estimada $\underline{\mathbf{G}}^{(M)}$. Para tener en cuenta la libertad rotacional, la core matrix es contra-rotada basándonos en las rotaciones ortogonales realizadas a las matrices de componentes.

El algoritmo $T3$-PCA

En la tabla 6.2 se presentan el valor medio del estadístico GOR para $\mathbf{A}$, $\mathbf{B}_1$, $\mathbf{C}$ y $\mathbf{B}_2$ y la media $MAD_{\underline{\mathbf{G}}}$, calculada para todos los análisis y para cada uno de los valores de α.

En esta tabla podemos observar como, de media, la recuperación de $\mathbf{B}_1$ y $\mathbf{C}$ es casi perfecta mientras que la de $\mathbf{A}$ y $\mathbf{B}_2$ es bastante buena (i.e., un valor ϕ mayor de 0,85 indica una buena recuperación mientras que un valor de ϕ mayor de 0,95 indica una recuperación excelente, (Lorenzo-Seva and ten Berge, 2006)).

Cuando analizamos los resultados para los distintos valores de α, parece que la recuperación de todos los parámetros se incrementa cuando el valor de alfa va de $0,050$ hasta $0,500$. Sin embargo para valores de α mayores, la recuperación de $\mathbf{A}$ y de $\mathbf{B}_2$ se decrementa, mientras que sigue siendo muy buena para $\mathbf{B}_1$ y $\mathbf{C}$.

Valor de alfa	$GOR_{\mathbf{A}}$	$GOR_{\mathbf{B}_1}$	$GOR_{\mathbf{C}}$	$MAD_{\underline{\mathbf{G}}}$	$GOR_{\mathbf{B}_2}$
0,050	0,85	0,95	0,96	941,11	0,79
0,100	0,93	0,91	0,97	950,78	0,80
0,300	0,96	0,96	0,99	1.061,93	0,90
0,500	**0,98**	**0,96**	**0,99**	**1.112,49**	**0,93**
0,600	0,96	0,96	0,99	1.130,49	0,93
0,700	0,94	0,96	0,99	1.144,75	0,92
0,800	0,92	0,96	0,99	1.154,13	0,91
0,900	0,90	0,96	0,99	1.159,02	0,90
0,950	0,88	0,96	0,99	1.160,23	0,90
0,999	0,88	0,96	0,99	1.160,69	0,83
Media global	0,91	0,96	0,99	1.097,56	0,89

Tabla 6.2: Recuperación media para las matrices de componentes (GOR) y para la core matrix (MAD) calculada para todos los conjuntos de datos y para cada uno de los valores de α.

Para estudiar como varía el rendimiento en la recuperación en función de los cinco factores del diseño y del valor de alfa, se llevan a cabo cinco análisis de la varianza $RMANOVA$, uno para cada medida de recuperación, donde la medida del rendimiento

es la variable dependiente y los cinco factores y el valor de alfa son las variables independientes. Sólo vamos a tener en cuenta los efectos con $\eta^2_G \geq 0{,}35$ (Bakeman, 2005).

La mayor variabilidad en $GOR_{\mathbf{A}}$ y $GOR_{\mathbf{B}_2}$ está explicada por el efecto principal importancia de la componente extra adicional ($\eta^2_G = 0{,}73$ y $\eta^2_G = 0{,}65$ respectivamente): el rendimiento en la recuperación disminuye cuando la importancia de la componente extra es mayor en términos de varianza explicada (Figura 6.2).

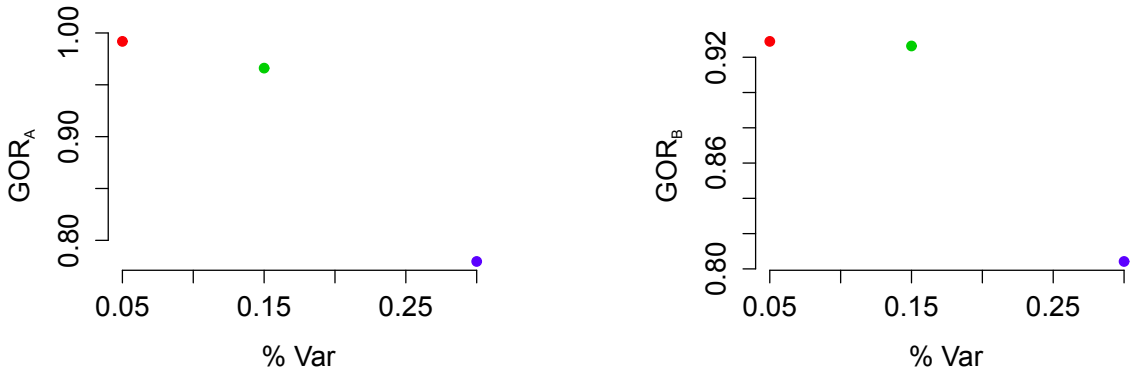


Figura 6.2: Valor medio de $GOR_{\mathbf{A}}$ (panel izquierdo) y $GOR_{\mathbf{B}_2}$ (panel derecho) según la importancia de la componente extra adicional.

Este efecto principal, está cuantificado por una considerable interacción entre el valor de α y la importancia de la componente distintiva ($\eta^2_G = 0{,}54$ y $\eta^2_G = 0{,}71$ respectivamente). En concreto, como se puede ver en la Figura 6.3, cuando la componente distintiva no es muy fuerte (i.e., 5 % y 15 % de variabilidad explicada), la recuperación se decrementa cuando se incrementa α. Además, la diferencia en la recuperación para datos con una componente distintiva no importante frente a aquellos que tienen una componente distinta media o fuerte se incrementa cuando α crece. Sin embargo, cuando existe una componente distintiva fuerte (i.e., explicando el 30 %) la recuperación es óptima para valores de α intermedios y empeora a media que nos acercamos a 0 (i.e., cuando todo el peso está en el bloque de dos vías) y 1 (i.e., sólo el bloque de tres vías tiene importancia en el análisis).

Para $GOR_{\mathbf{B}_1}$ y $GOR_{\mathbf{C}}$ no existen resultados significativos.

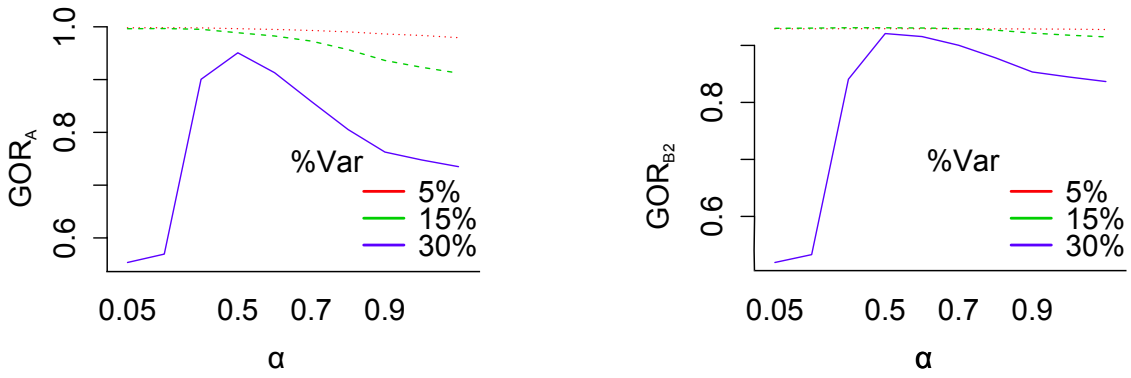


Figura 6.3: Valor medio de $GOR_{\mathbf{A}}$ (panel izquierdo) y $GOR_{\mathbf{B}_2}$ (panel derecho) en función de α y de la importancia de la componente distintiva.

Dado que para alfa igual a 0,5 (i.e., ambos bloques tienen la misma influencia en el análisis) la recuperación parece óptima, vamos a continuar el análisis con $\alpha = 0,5$.

Para investigar como depende de los cinco factores considerados el rendimiento, se realiza un análisis de la varianza ($ANOVA$) siendo la variable dependiente cada rendimiento en la recuperación y las variables independientes los cinco factores manipulados. Sólo vamos a tener en cuenta efectos principales para los cuales el coeficiente de correlación intraclase $\hat{p}_I$ (Haggard, 1958; Kirk, 1982) sea mayor que 0,05.

El rendimiento en la recuperación de la matriz de componentes **A** (Figura 6.4) es menor cuando la componente extra es más importante ($\hat{p}_I = 0,39$). Así el valor de $GOR_{\mathbf{A}}$ es igual a $0,996$; $0,989$ y $0,951$ cuando la componente distintiva es débil, media y fuerte, respectivamente. Además existe interacción entre el rango del modelo y la cantidad de variabilidad explicada por la componente extra ($\hat{p}_I = 0,09$). El decremento en $GOR_{\mathbf{A}}$ para valores crecientes de la importancia de la componente distintiva es menos pronunciado cuando el rango es mayor (Figura 6.5).

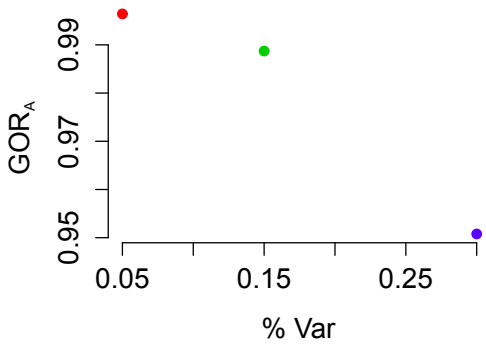


Figura 6.4: Valor medio de $GOR_{\mathbf{A}}$ según la importancia de la componente extra adicional

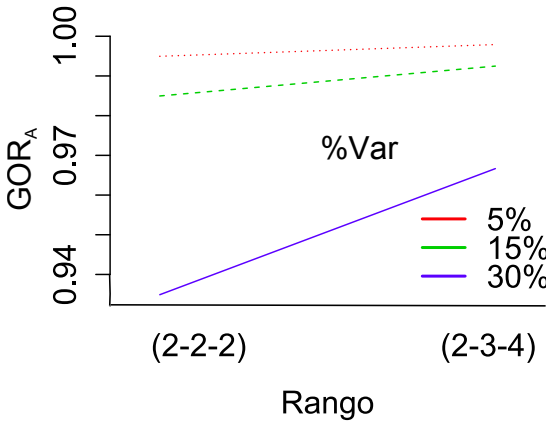


Figura 6.5: Valor medio de $GOR_{\mathbf{A}}$ como función del rango del modelo y de la importancia de la componente extra

El rendimiento en la recuperación de $\mathbf{B}_1$ se incrementa cuando se incrementa el rango ($\hat{p}_I = 0{,}14$; valor medio $GOR_{\mathbf{B}_1}$ de 0,951 y 0,965 para rango $(2,2,2)$ y $(2,3,4)$, respectivamente) y cuando aumenta la cantidad de ruido en los datos ($\hat{p}_I = 0{,}21$; valor medio $GOR_{\mathbf{B}_1}$ de 0,950 y 0,966 para ε igual a 0,15 y 0,30; respectivamente).

Diferencias en recuperación entre el algoritmo $T3$-PCA y la estrategia segmentada

En este apartado, compararemos la estrategia integrada $T3$-PCA y la estrategia segmentada (i.e., primero ajustando un modelo $Tucker3$ al array de tres vías y después ajustando un PCA a la matriz de dos vías bajo la restricción de que la matriz de componentes $\mathbf{A}$ ha de ser igual) en términos de cómo descubren las matrices de componentes y la core matrix. Para ello se calcula para cada medida de recuperación y para cada cada análisis, la diferencia en la recuperación entre la estrategia integrada y la estrategia segmentada (e.g., para $\mathbf{A}$, $GOR_{\mathbf{A}}^{diff} = GOR_{\mathbf{A}}^{T3-PCA} - GOR_{\mathbf{A}}^{segmented}$).

En la Tabla 6.3 se presenta el valor de la diferencia media en la recuperación para cada una de las medidas calculado para todos los análisis y para cada valor de α.

A partir de esta tabla, se puede ver que la estrategia segmentada supera a $T3$-PCA cuando la información en el bloque de dos vías domina el análisis (i.e., $\alpha \leq 0,10$). Sin embargo, cuando la influencia de los dos bloques en el análisis está más balanceada, funciona mejor $T3$-PCA que la estrategia segmentada. En concreto, esto es especialmente cierto para la recuperación del modo común $\mathbf{A}$ y en menor medida para $\mathbf{B}_2$, mientras que no se encuentran diferencias para $\mathbf{B}_1$ y $\mathbf{C}$. Cuando el bloque de datos de tres vías domina el análisis, ambas estrategias funcionan igual. La mayor diferencia se observa cuando ambos bloques de datos tienen la misma influencia en el análisis (i.e., $\alpha = 0,50$).

Para estudiar como varía la diferencia en el rendimiento en función de los cinco factores y del valor de alfa, se realizan cinco análisis de la varianza $RMANOVA$, donde la diferencia en rendimiento es la variable dependiente y los cinco factores junto con el valor de alfa son las variables independientes. Sólo se tienen en cuenta efectos con $\eta_G^2 \geq 0{,}35$ (Bakeman, 2005).

El algoritmo $T3$-PCA supera a la estrategia segmentada en la recuperación de $\mathbf{A}$ y $\mathbf{B}_2$ cuando la información en ambos bloques de datos se pondera (más o menos) de igual forma (η_G^2 igual a $0,33$ y $0,48$ para $\mathbf{A}$ y $\mathbf{B}_2$). Como se puede ver en la Tabla 6.3 el mayor valor para $GOR_{\mathbf{A}}^{diff}$ y $GOR_{\mathbf{B}_2}^{diff}$ se da para valores intermedios de α (i.e., alrededor de

Valor de alfa	$GOR^{diff}_{\mathbf{A}}$	$GOR^{diff}_{\mathbf{B}_1}$	$GOR^{diff}_{\mathbf{C}}$	$MAD^{diff}_{\mathbf{G}}$	$GOR^{diff}_{\mathbf{B}_2}$
0,050	-0,03	-0,08	-0,03	-219,59	-0,10
0,100	-0,02	-0,01	-0,03	-209,91	-0,10
0,300	0,09	0,00	-0,03	-98,76	0,01
0,500	**0,10**	**0,00**	**0,00**	**-48,20**	**0,03**
0,600	0,09	0,00	0,00	-30,20	0,03
0,700	0,07	0,00	0,00	-15,94	0,03
0,800	0,04	0,00	0,00	-6,57	0,02
0,900	0,01	0,00	0,00	-1,68	0,01
0,950	0,01	0,00	0,00	-0,46	0,00
0,999	0,00	0,00	0,00	0,00	0,00
Media global	0,04	0,00	-0,01	-63,13	-0,01

Tabla 6.3: Diferencia en la recuperación media entre la estrategia integrada $T3\text{-}PCA$ y la estrategia segmentada para las matrices de componentes (GOR^{diff}) y para la core matrix (MAD^{diff}) calculada para todos los conjuntos de datos y para cada uno de los valores de α.

0,50). Además, esta diferencia en la recuperación de ambas matrices de componentes se decrementa cuando uno de los bloques de datos domina el análisis, siendo este efecto más fuerte cuando es el bloque de dos vías (i.e., llegando a tomar el valor medio de GOR^{diff} valores negativos cuando $\alpha \leq 0,10$). Este efecto principal, sin embargo, está cuantificado por una interacción entre la importancia de la componente extra adicional y α (η^2_G es $0,48$ y $0,66$ para $\mathbf{A}$ y $\mathbf{B}_2$).

En la Figura 6.6, se puede ver que $T3\text{-}PCA$ siempre recupera el modo común mejor que la estrategia segmentada cuando $\alpha \geq 0,50$, siendo la diferencia entre ambas estrategias mayor cuando la componente distintiva es más importante. Para valores bajos de α, sin embargo, el algoritmo $T3\text{-}PCA$ sólo supera a la estrategia segmentada cuando la componente distintiva no es demasiado fuerte.

Dado que hemos visto que las mayores diferencias se dan cuando los dos bloques de datos tienen igual importancia en el análisis, nos vamos a centrar en el análisis para $\alpha = 0,50$. Para estudiar cómo influyen los factores manipulados en la diferencia en la recuperación, se llevan a cabo cinco $ANOVA$s, siendo la variable dependiente la diferencia en la recuperación y las variables independientes los factores del diseño. Sólo

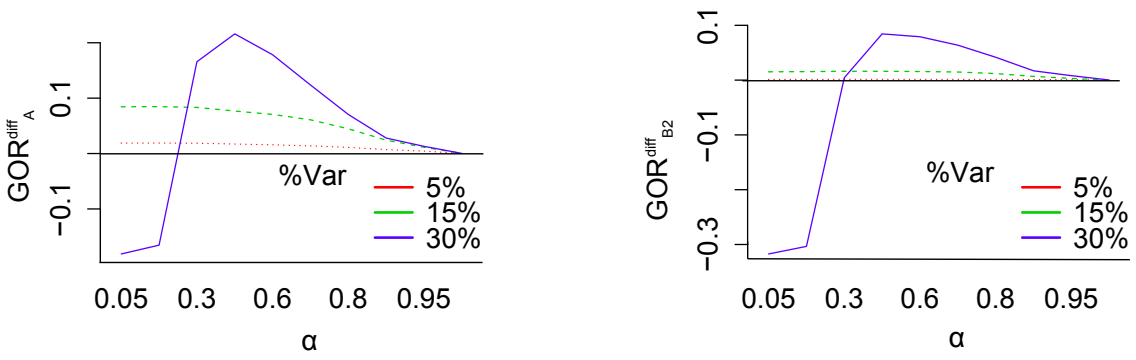


Figura 6.6: Valor medio de $GOR_{\mathbf{A}}^{diff}$ (panel izquierdo) y $GOR_{\mathbf{B}_2}^{diff}$ (panel derecho) en función de α y de la importancia de la componente distintiva.

vamos a tener en cuenta efectos principales para los cuales el coeficiente de correlación intraclase $\hat{p}_I > 0,10$ (Haggard, 1958; Kirk, 1982).

Se observa como las diferencias en la recuperación de $\mathbf{A}$ y $\mathbf{B}_2$ se deben fundamentalmente a la importancia de la componente distintiva y del rango del modelo. En la Tabla 6.4, se muestra el valor medio de $GOR_{\mathbf{A}}^{diff}$ y $GOR_{\mathbf{B}_2}^{diff}$ como función del rango del modelo y de la importancia de la componente distintiva.

Como se puede observar en la Tabla, parece que $T3$-PCA supera a la estrategia segmentada en mayor medida cuando la componente distintiva se hace más influyente ($\hat{p}_I$ es $0,42$ y $0,24$ para $\mathbf{A}$ y $\mathbf{B}_2$). Además, existe una considerable interacción entre el rango del modelo y la importancia de la componente distintiva ($\hat{p}_I$ igual a $0,15$ y $0,11$ para $\mathbf{A}$ y $\mathbf{B}_2$). Como se puede ver en la Tabla, cuando se incrementa el rango y la componente distintiva es débil o intermedia la diferencia se decrementa, pero cuando se incrementa el rango y existe una componente distintiva fuerte, la diferencia en la recuperación se incrementa a favor de la estrategia integrada.

	$GOR_{\mathbf{A}}^{diff}$			$GOR_{\mathbf{B}_2}^{diff}$		
Rank of the model	5 %	15 %	30 %	5 %	15 %	30 %
$(2, 2, 2)$	0,029	0,107	0,174	0,003	0,028	0,061
$(2, 3, 4)$	0,005	0,046	0,259	0,000	0,004	0,108
Overall	0,017	0,077	0,216	0,001	0,016	0,084

Tabla 6.4: Valor medio de $GOR_{\mathbf{A}}^{diff}$ y $GOR_{\mathbf{B}_2}^{diff}$ como función del rango del modelo y la importancia de la componente distintiva ($\alpha = 0{,}50$).

DISCUSIÓN DE LOS RESULTADOS

Los resultados del estudio demuestran que el algoritmo $T3$-PCA optimiza de forma adecuada la función de pérdida 6.3 cuando se toma como solución inicial la mejor de entre una solución inicial racional y 50 soluciones aleatorias.

En lo relativo a la recuperación, parece que el algoritmo $T3$-PCA recupera bien las componentes comunes. Para las componentes no comunes, se observa que los modos del modelo $Tucker3$ se recuperan mejor que el modo del modelo PCA. Los mejores resultados se obtienen cuando los dos bloques de datos tienen la misma influencia en el análisis (i.e., $\alpha = 0,5$) y especialmente cuando existe una componente distintiva fuerte. Más áun, en este último caso, la recuperación disminuye cuando uno de los dos bloques domina el análisis, siendo este efecto más pronunciado cuando es el bloque PCA el que domina el análisis (Figura 6.2).

Cuando se comparan la estrategia integrada $T3-PCA$ con la estretegia segmentada, parece que en general, la estrategia integrada recupera mejor el modo común y en menor medida también el modo PCA no común, mientras que no existen diferencias entre ambas estrategias en la recuperación de los modos $Tucker3$ no comunes.

Las mayores diferencias en la recuperación se dan para $\alpha = 0,5$. Cuando existe una componente distintiva fuerte, la diferencia en la recuperación disminuye a media que nos aproximamos a 0 ó a 1(i.e., cuando domina la parte PCA o $Tucker3$, respectivamente). Cuando la parte PCA domina el análisis, la estrategia segmentada incluso supera a la integrada (Figura 6.6).

Como conclusión podemos afirmar que si conocemos el rango óptimo del modelo $T3 - PCA$, el algoritmo $T3 - PCA$ obtiene buenas soluciones cuando se utiliza una inicialización racional y 50 aleatorias y se dan valores de α intermedios.

7

SOFTWARE R

En este capítulo se incluye el software necesario para llevar a cabo el análisis de un modelo $T3$-PCA así como el software que se ha utilizado para llevar a cabo las simulaciones explicadas en el Capítulo 6.

Son varios programas desarrollados en el lenguaje R versión (3.0.2). A continuación se enumeran los mas importantes junto con una breve descripción de para qué sirven cada uno de ellos.

1. $PreProcess3D.R$: Se encarga del preprocesamiento de la matriz de tres vías

2. $T3PCAfunc.R$: Se encarga del análisis T3-PCA de una matriz de tres vías y una matriz de dos vías que están acopladas por un modo.

3. $T3PCAsegmented.R$: Se encarga de analizar una matriz de tres vías y una matriz de dos vías que están acopladas por un modo, mediante una estrategia segmentada. El tipo de estrategia depende de un parámetro: SegmentedType. Cuando SegmentedType es 1 aplica un $Tucker3$ a la matriz de tres vías y después un PCA a la matriz de dos vías con la restricción de que la matriz de componentes $\mathbf{A}$ del modo común ha de ser igual en ambos modelos. Cuando SegmentedType es igual a 0, aplica en primer lugar un PCA a la matriz de dos vías y a continuación un $Tucker3$ a la matriz de tres vías con la restricción de que la matriz de componentes $\mathbf{A}$ del modo común ha de ser igual en ambos modelos.

4. $T3PCAseparate.R$: Se encarga de analizar una matriz de tres vías y una matriz de dos vías que están acopladas por un modo, mediante una estrategia separada. Por tanto aplica de forma independiente un $Tucker3$ a la matriz de tres vías y un PCA a la matriz de dos vías.

5. $ComputeRecoveryT3PCA.R$: A partir de la solución real (sin componente extra adicional) y de la solución obtenida a partir de cualquiera de los algoritmos anteriores, este procedimiento se encarga de calcular el Recovery de las matrices de componentes y de la core matrix.

6. $ComputeRecoveryT3PCAExtra.R$: Calcula las congruencias entre la solución real (cuando se ha añadido una componente extra adicional) y la solución obtenida a partir de cualquiera de los algoritmos anteriores.

7. $GenerateTucker3-PCA.R$: Genera de forma aleatoria las matrices de componentes y la core matrix. A partir de estas construye el modelo real y posteriormente le añade ruido.

8. $doTucker3PCAsimulation.R$: Realiza el estudio de simulación que se ha detallado en el capítulo 6.

7.1. PREPROCESAMIENTO DE DATOS: PrePro-cess3D.R

```
PreProcess3D <- function(Ar) {
    # pre-processing of a 3D-matrix take person (first mod) by variable
    # (second mode) by situation (third mode) data for each variable-
    # situation combination, the mean over persons is zero for each variable
    # (over personsand situations) the variance equals one

    # Ar (i x j x k): 3D array

    nRow = dim(Ar)[1]
    nCol = dim(Ar)[2]
    nSlice = dim(Ar)[3]

    MeansMatrix = matrix(0, nCol, nSlice)
    for (coltel in 1:nCol) {
        for (slicetel in 1:nSlice) {
            MeansMatrix[coltel, slicetel] = mean(Ar[, coltel, slicetel])
        }
    }

    Ar_pp = array(0, cbind(nRow, nCol, nSlice))
    for (rowtel in 1:nRow) {
        for (coltel in 1:nCol) {
            for (slicetel in 1:nSlice) {
                Ar_pp[rowtel, coltel, slicetel] = Ar[rowtel, coltel, slicetel]
                - MeansMatrix[coltel, slicetel]
            }
        }
```

```
    }

    VarMatrix = matrix(0, 1, nCol)
    for (coltel in 1:nCol) {
        VarMatrix[coltel] = var(matrix(Ar_pp[, coltel, ], nRow * nSlice, 1))
    }

    for (rowtel in 1:nRow) {
        for (coltel in 1:nCol) {
            for (slicetel in 1:nSlice) {
                Ar_pp[rowtel, coltel, slicetel] = Ar_pp[rowtel, coltel,
                slicetel]/sqrt(VarMatrix[coltel])
            }
        }
    }

    return(Ar_pp)
}
```

7.2. ANALIZA DATOS

7.2.1. ALGORITMO T3-PCA: T3PCAfunc.R

En la figura 7.1 se muestra un organigrama de la función **T3PCAfunc.R**

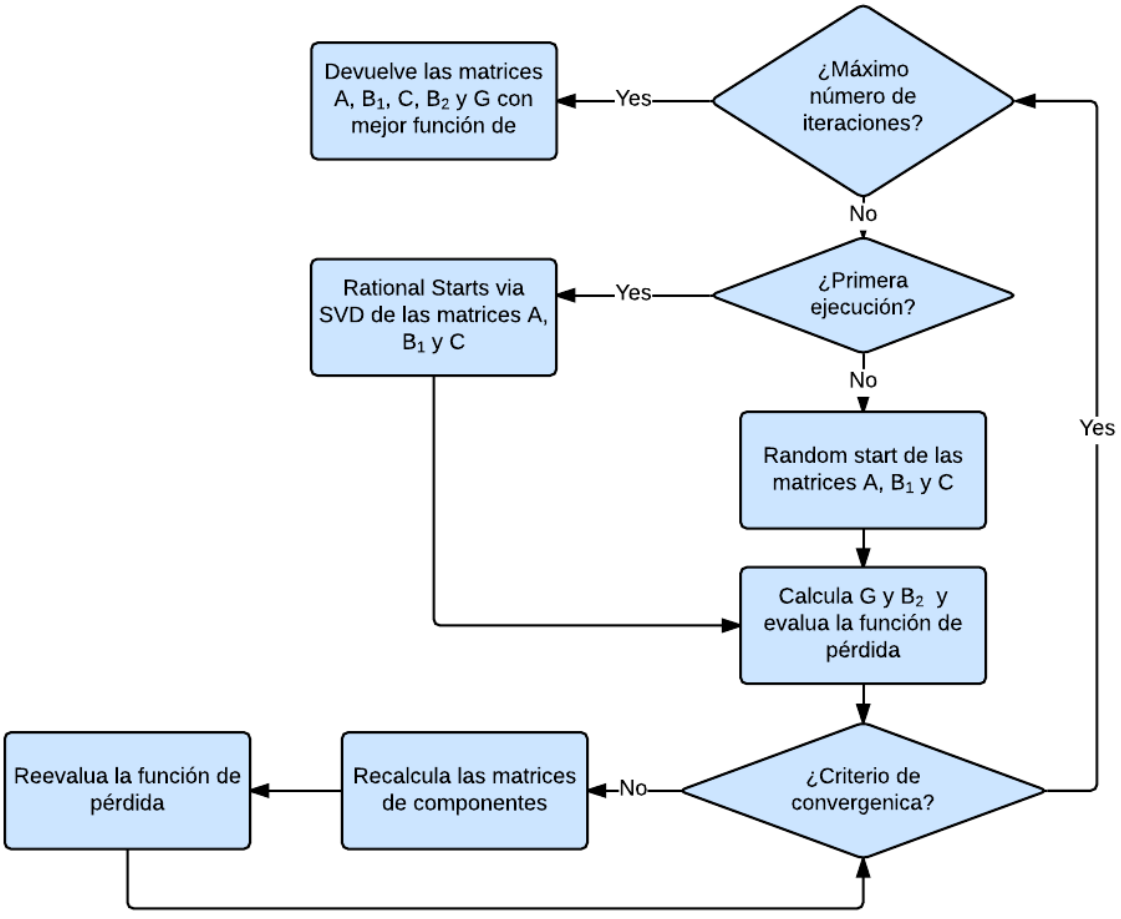


Figura 7.1: Organigrama del procedimiento *T3-PCA*.

```
T3PCAfunc <- function(X3D, Y, n, m, p, q, r1, r2, r3, conv,
OriginalAlfa, AlternativeLossF, nRuns, StartSeed )
{
    # INPUT X3D (n x m x p): 3D-data matrix (represented as a 3D-array)
    # Y (n x q): 2D-data matrix
    # n: number of elements of first mode of 3D/2D (the common mode: rows)
    # m: number of elements of second mode of 3D (columns 3D)
    # p: number of elements of third mode of 3D (slabs)
    # q: number of elements of second mode of 2D (columns 2D)
    # r1,r2,r3: rank of the model Tucker3-model
    # conv: value for convergence (tolerance value)
    # OriginalAlfa (0-1): weight for the 3D-block (1-alfa is weight
    # for the 2D-block
    # AlternativeLossF: using the alternative loss function?
    # 0 = no (use original loss function: weighted SSQ; weighted met alfa)
    # 1 = yes (use weighted loss function with scaled SSQ: scaled by the
    # SSQ in X and y)
    # nRuns: number of runs StartSeed: seed for the analysis
    # OUTPUT Sol: structure for the solution containing the following
    # information - Info * AnalysisType * nRows * nColumns3D * nSlices *
    # nColumns2D * RankVector * TolPercentage * OriginalAlfa * Alfa *
    # AlternativeLossF * nRuns * StartSeed - A - B1 - C - H - B2 - LossWeighted
    # - LossUnweighted - FitPercentage - FitPercentage3D - nIter - FitValues -
    # nIterValues - La - Lb - Lc - Fit3D - Fit2D - Fit - Fit3Dsize - Fit2Dsize -
    # Fitsize - CpuTime - TimeSeconds

    ## take scaling into account ###

    set.seed(StartSeed)
    source("ComputeTucker3PCAloss.R")
    library(ThreeWay, MASS)
```

```
checkinput = 1
if (r1 > min(n, m * p, q)) {
    cat(" ", fill = TRUE)
    cat("rank1 should be an integer between 1 and ",
     min(i, l), fill = TRUE)
    cat(" ", fill = TRUE)
    checkinput = 0
}

if (r2 > min(m, n * p)) {
    cat(" ", fill = TRUE)
    cat("rank2 should be an integer between 1 and ",
     min(j, i * k), fill = TRUE)
    cat(" ", fill = TRUE)
    checkinput = 0
}

if (r3 > min(p, n * m)) {
    cat(" ", fill = TRUE)
    cat("rank3 should be an integer between 1 and ",
     min(k, i * j), fill = TRUE)
    cat(" ", fill = TRUE)
    checkinput = 0
}

if ((r1 > r2 * r3) | (r2 > r1 * r3) | (r3 > r1 * r2)) {
    cat(" ", fill = TRUE)
    cat("None of the ranks can be larger than the products
     of the other two (e.g., rank1 > rank2*rank3 is
     not allowed", fill = TRUE)
    cat(" ", fill = TRUE)
    checkinput = 0
```

```
}

if ((OriginalAlfa < 0) || (OriginalAlfa >= 1)) {
    cat(" ", fill = TRUE)
    cat("OriginalAlfa should be between 0 and 1
     (but not 0 or 1)", fill = TRUE)
    cat(" ", fill = TRUE)
    checkinput = 0
}

if ((AlternativeLossF != 0) && (AlternativeLossF != 1)) {
    cat(" ", fill = TRUE)
    cat("AlternativeLossF should be 0 or 1", fill = TRUE)
    cat(" ", fill = TRUE)
    checkinput = 0
}

if (checkinput == 1) {
    X = matrix(X3D, n, m * p)

    # P slices are concatenated horizontally

    ssq3D = sum(X^2)
    ssq2D = sum(Y^2)

    if (AlternativeLossF == 1) {
        Alfa = (OriginalAlfa * ssq2D)/((OriginalAlfa * ssq2D)
         + ((1 - OriginalAlfa) * ssq3D))
    } else {
        Alfa = OriginalAlfa
    }
```

```
BestA = matrix(0, n, r1)
BestB1 = matrix(0, m, r2)
BestC = matrix(0, p, r3)
BestH = array(0, cbind(r1, r2, r3))  # CoreArray
BestB2 = matrix(0, q, r1)
BestIter = -9999
BestLoss = 999999999999

cputime = system.time({
    FitValues = matrix(0, 1, nRuns + 1)
    nIterValues = matrix(0, 1, nRuns + 1)
```

Dado que los algoritmos ALS pueden caer en un mínimo local, la idea es ejecutar el algoritmo varias veces (nRuns veces), cada vez con un valor inicial diferente, y retener el modelo para el cual la función de pérdida es menor.

La primera vez se inicializan las matrices A, B_1 y C mediante la descomposición en valores singulares. El resto de las veces se inicializarán de forma aleatoria.

Una vez inicializadas las matrices anteriores se calculan la core matrix $\underline{G}$ y B_2 y se evalua el valor de la función de pérdida.

```
    for (run in 1:nRuns + 1) {

        # initialize A, B1 and C (orthonormal)

        if (run == 1) {

          # rational starts via eigendecomposition
           (gives orthonormal starting values)

          EIG = eigen(cbind(X, Y) %*% t(cbind(X, Y)))
```

```
  A = EIG$vectors[, 1:r1]
  rm(EIG)

  Z = permnew(X, n, m, p)  # yields m x p x n array
  EIG = eigen(Z %*% t(Z))
  B1 = EIG$vectors[, 1:r2]
  rm(EIG)

  Z = permnew(Z, m, p, n)  # yields p x n x m array
  EIG = eigen(Z %*% t(Z))
  C = EIG$vectors[, 1:r3]
  rm(EIG, Z)
} else {

  # random start (orthonormal)

  A = orth(matrix(rnorm(n * r1, 0, 1), n, r1))
  B1 = orth(matrix(rnorm(m * r2, 0, 1), m, r2))
  C = orth(matrix(rnorm(p * r3, 0, 1), p, r3))
}

# Calculate initial Core

Z = permnew(t(A) %*% X, r1, m, p)
Z = permnew(t(B1) %*% Z, r2, p, r1)
H = permnew(t(C) %*% Z, r3, r1, r2)

# H (r1 x r2r3)

rm(Z)

# Calcule initial B2 (is in general not orthogonal!)
```

```
B2 = t(solve(t(A) %*% A) %*% t(A) %*% Y)
model2D = A %*% t(B2)

# Evaluate f

f = ComputeTucker3PCAloss(A, B1, C, H, B2, X, Y,
 Alfa)
```

Mientras no se alcance el criterio de convergencia se recalculan las matrices de componentes A, $\mathbf{B}_1$, C, $\mathbf{B}_2$, y la core matrix $\underline{\mathbf{G}}$.

Las matrices de componentes $\mathbf{B}_1$ y $\mathbf{C}$ y la core matrix $\underline{\mathbf{G}}$ se modifican según el algoritmo TUCKALS3:

$$\mathbf{B}_1 = \mathbf{D}_b^1(\mathbf{C} \otimes \mathbf{A})$$

$$\mathbf{C} = \mathbf{D}_c^1(\mathbf{B}_1 \otimes \mathbf{A})$$

$$\mathbf{G}_a = \mathbf{A}^{'}\mathbf{D}_a^1(\mathbf{B}_1 \otimes \mathbf{C})$$

donde $\mathbf{G}_a$ es la $I \times JK$ version matriciada de $\underline{\mathbf{G}}$, mientras que la actualización de $\mathbf{B}_2$ es un problema de regresión múltiple multivariante:

$$\mathbf{B}_2 = (\mathbf{A}^{'}\mathbf{A})^{-1}\mathbf{A}^{'}\mathbf{D}^2$$

y $\mathbf{A}$ se actualiza como sigue:

$$\mathbf{A} = \mathbf{U}\mathbf{V}^{'}$$

donde $\mathbf{U}$ es el vector singular izquierdo de $\mathbf{Y}\mathbf{Z}^{'}$ y $\mathbf{V}$ es el vector singular derecho de $\mathbf{Y}\mathbf{Z}^{'}$, con

$$\mathbf{Y} = \sqrt{\alpha} \times \mathbf{D}_a^1 | \sqrt{(1-\alpha)} \times \mathbf{D}^2$$

$$\mathbf{Z} = \sqrt{\alpha} \times \mathbf{G}_a(\mathbf{C}^{'} \otimes \mathbf{B}_1^{'}) | \sqrt{1-\alpha} \times \mathbf{B}_2^{'}$$

```
iter = 0
fold = f + (2 * conv * f)
V = cbind((sqrt(Alfa) * X), (sqrt(1 - Alfa) * Y))
while ((fold - f) > (f * conv)) {
  iter = iter + 1
  fold = f

  # update A (orthonormal)

  W = cbind((sqrt(Alfa) * H %*% kronecker(t(C), t(B1))),
   (sqrt(1 - Alfa) * t(B2)))
  Asvd = svd(V %*% t(W))
  A = Asvd$u %*%
  t(Asvd$v)

  # is orthogonal update A'A = I

  rm(W, Asvd)

  # update B1 (orthonormal)

  Z = permnew(X, n, m, p)
  Z = permnew(Z, m, p, n)
  Z = permnew(t(C) %*% Z, r3, n, m)
  Z = permnew(t(A) %*% Z, r1, m, r3)

  # yields m x r3 x r1 array

  B1 = qr.Q(qr(Z %*% (t(Z) %*% B1)), complete = FALSE)
```

```
  rm(Z)

  # update C (orthonormal)

  Z = permnew(t(A) %*% X, r1, m, p)
  Z = permnew(t(B1) %*% Z, r2, p, r1)

  # yields p x r1 x r2 array

  C = qr.Q(qr(Z %*% (t(Z) %*% C)), complete = FALSE)
  rm(Z)

  # Update H (Core)

  Z = permnew(t(A) %*% X, r1, m, p)
  Z = permnew(t(B1) %*% Z, r2, p, r1)
  H = permnew(t(C) %*% Z, r3, r1, r2)
  rm(Z)

  # Update B2 (not necessarily orthogonal !!)

  B2 = t(solve(t(A) %*% A) %*% t(A) %*% Y)
  Model2D = A %*% t(B2)

  # Evaluate f

  f = ComputeTucker3PCAloss(A, B1, C, H, B2, X, Y, Alfa)
}  #end of while-loop (alternating part of the algorithm)

FitValues[run] = f
nIterValues[run] = iter
```

Se retiene la solución con menor valor para la función de pérdida de entre las obtenidas en las nRuns ejecuciones

```
        if (f < BestLoss) {
          BestA = A
          BestB1 = B1
          BestC = C
          BestB2 = B2
          BestH = H
          BestLoss = f
          BestIter = iter
        }
    }  # end of for-loop (nRuns)
})  # end of system.time

# compute 'intrinsic eigenvalues'

La = BestH %*% t(BestH)
J = permnew(BestH, r1, r2, r3)
Lb = J %*% t(J)
J = permnew(J, r2, r3, r1)
Lc = J %*% t(J)

# compute BOF

Model3D = BestA %*% BestH %*% kronecker(t(BestC), t(BestB1))
Model2D = BestA %*% t(BestB2)
BOF3D = sum((Model3D - X)^2)
BOF2D = sum((Model2D - Y)^2)

FitPercentage = ((Alfa * (sum(BestH^2)/ssq3D)) + ((1 - Alfa) *
 (sum(Model2D^2)/ssq2D))) * 100
```

```
FitPercentage3D = 100 * ((ssq3D - BOF3D)/ssq3D)

out = list()
out$Info = list()
out$Info$AnalysisType = "simultaneous"
out$Info$nRows = n
out$Info$nColumns3D = m
out$Info$nSlices = p
out$Info$nColumns2D = q
out$Info$RankVector = cbind(r1, r2, r3)
out$Info$TolPercentage = conv
out$Info$OriginalAlfa = OriginalAlfa
out$Info$Alfa = Alfa
out$Info$AlternativeLossF = AlternativeLossF
out$Info$nRuns = nRuns
out$Info$StartSeed = StartSeed
out$A = BestA
out$B1 = BestB1
out$C = BestC
out$H = BestH
out$B2 = BestB2
out$LossWeighted = BestLoss
out$LossUnweighted = BOF3D + BOF2D
out$FitPercentage = FitPercentage
out$FitPercentage3D = FitPercentage3D
out$nIter = BestIter
out$FitValues = FitValues
out$nIterValues = nIterValues
out$La = La
out$Lb = Lb
out$Lc = Lc
out$Fit3D = BOF3D
```

```
        out$Fit2D = BOF2D
        out$Fit = (Alfa * BOF3D) + ((1 - Alfa) * BOF2D)
        out$Fit3Dsize = BOF3D/(n * m * p)
        out$Fit2Dsize = BOF2D/(n * q)
        out$Fitsize = out$Fit/((n * m * p) + (n * q))
        out$CpuTime = cputime[1]
        out$TimeSeconds = round(cputime[1], 2)
        return(out)
    }
}
```

Para calcular la función de pérdida se utiliza la función **ComputeTucker3PCAloss.R**

```
ComputeTucker3PCAloss <- function(A, B1, C, Hmatrix, B2, Xdata, Ydata,
Alfa) {
    # A (i x r1): component matrix for 1st mode of 3D (common mode)
    # B1 (j x r2): component matrix for 2nd mode of 3D
    # C (k x r3): component matrix for 3th mode of 3D
    # Hmatrix (r1 x r2r3): CoreArray (in matrix form)
    # B2 (l x r1): component matrix for 2nd mode of 2D
    # Xdata (i x jk): 3D data matrix
    # Ydata (i x l): 2D data matrix
    # Alfa (0-1): weight for the 3D part

    Model3D = A %*% Hmatrix %*% kronecker(t(C), t(B1))
    Model2D = A %*% t(B2)
    Loss = (Alfa * sum((Xdata - Model3D)^2))
    + ((1 - Alfa) * sum((Ydata - Model2D)^2))
    return(Loss)
}
```

7.2.2. ESTRATEGIA SEGMENTADA: T3PCAsegmented.R

Tanto la función **T3PCAsegmented.R** como la función **T3PCAseparate.R** utilizan la función **T3funcSeed**. Esta función dado un array de tres vías, ajusta un modelo Tucker3 donde las matrices **A**, **B** y **C** son ortonormales.

En la Figura 7.2 se muestra un organigrama del procedimiento.

```
T3funcSeed <- function ( X , n , m , p , r1 , r2 , r3 , start
, conv , StartSeed , StartA , StartB , StartC , StartH )
{
  # fits a Tucker3 model with orthonormal A, B and C

  # X (n x mp): matricized 3D-data array (frontal slices)
  # n: number of elements of first mode (the common mode: rows)
  # m: number of elements of second mode (columns)
  # p: number of elements of third mode (slabs)
  # r1,r2,r3: rank of the model Tucker3-model
  # start: type of start for the analysis
  #        0 = rational start (from generalized eigenvalue decomposition:
  #         orthonormal component matrices)
  #        1 = random start (random orthonormalized component matrices)
  #        2 = user-specified initial component matrices
  #            (best to take orthonormal component matrices)
  # conv: convergence criterion (tolerance value)
  # StartSeed: seed to start the analysis
  #  [StartA, StartB, StartC, StartH: eventually one may specify initial
  #     A,B, C and H matrices (necessary if start=2)]
  #  [  StartA (n x r1): initial row component matrix]
  #  [  StartB (m x r2): initial column component matrix
  #  [  StartC (p x r3): initial slab component matrix
```

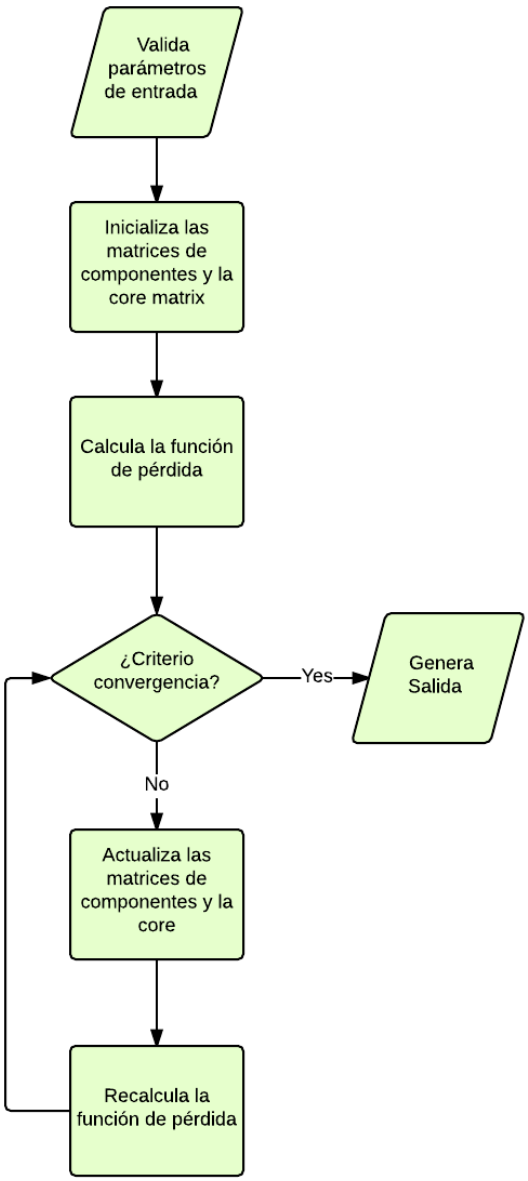


Figura 7.2: Organigrama del procedimiento **T3funcSeed**

```
#  [  StartH (r1 x r2r3): initial (matricized) core array (frontal slices)

set.seed( StartSeed ) # only for runif
```

```
library( ThreeWay )

checkinput = 1

if ( (start != 0) & (start != 1) & (start != 2) )
{
  cat(" ",fill=TRUE)
  cat("start should be 0 (rational start), 1 (random start)
  or 2 (user specified start)",fill=TRUE)
  cat(" ",fill=TRUE)
  checkinput=0
}

if ( length( dim(X) ) != 2 )
{
  X = matrix( X , n , m*p ) # P slices are concatenated horizontally
}

X = as.matrix(X)

if (checkinput == 1)
{
  cputime = system.time(
  {
    ss = sum( X ^ 2 )
```

Si el parámetro de entrada start es igual a 1 inicializa las matrices de componentes mediante la descomposición en valores singulares de XX' siendo X la matriciación horizontal del array de tres vías

```
if (start == 0)
{
  EIG = eigen(X %*% t(X))
  A = EIG$vectors[, 1:r1]
  Z = permnew(X, n, m, p)
  EIG = eigen(Z %*% t(Z))
  B = EIG$vectors[, 1:r2]
  Z = permnew(Z, m, p, n)
  EIG = eigen(Z %*% t(Z))
  C = EIG$vectors[, 1:r3]
}
else
{
```

Si start es igual a 1, las matrices de componentes se generan de forma aleatoria.

```
  if (start == 1)
  {
    if (n >= r1)
    {
      A = orth( matrix( runif( n * r1 , 0 , 1 ) , n , r1 ) - 0.5 )
    }
    else
    {
      A = orth( matrix( runif( r1 * r1 , 0 , 1 ) , r1 , r1 ) - 0.5 )
      A = A[1:n, ]
    }

    if (m >= r2)
    {
```

```
    B = orth( matrix( runif( m * r2 , 0 , 1 ) , m , r2 ) - 0.5 )
  }
  else
  {
    B = orth( matrix( runif( r2 * r2 , 0 , 1 ) , r2 , r2 ) - 0.5 )
    B = B[1:m, ]
  }

  if (p >= r3)
  {
    C = orth( matrix( runif( p * r3 , 0 , 1 ) , p , r3 ) - 0.5 )
  }
  else
  {
    C = orth( matrix( runif( r3 * r3 , 0 , 1 ) , r3 , r3 ) - 0.5 )
    C = C[1:p, ]
  }
}
else #user-specified value (should be orthonormal A, B and C)
{
```

Las matrices de componentes se reciben como parámetro de entrada.

```
  if ( sum( ( ( t(StartA) %*% StartA ) - diag(r1) ) ^ 2 ) > .0000001 )
  {
    A = orth( StartA )
  }
  else
  {
    A = StartA
  }
```

```
    if ( sum( ( ( t(StartB) %*% StartB ) - diag(r2) ) ^ 2 ) > .0000001 )
    {
      B = orth( StartB )
    }
    else
    {
      B = StartB
    }

    if ( sum( ( ( t(StartC) %*% StartC ) - diag(r3) ) ^ 2 ) > .0000001 )
    {
      C = orth( StartC )
    }
    else
    {
      C = StartC
    }

    H = StartH
  }
}


# Compute loss function value


if (start != 2)
{
  Z = permnew(t(A) %*% X, r1, m, p)
  Z = permnew(t(B) %*% Z, r2, p, r1)
  H = permnew(t(C) %*% Z, r3, r1, r2)
  f = ss - sum( H ^ 2 )
```

```
}
else
{
  Z = B %*% permnew(A %*% H, n, r2, r3)
  Z = C %*% permnew(Z, m, r3, n)
  Z = permnew(Z, p, n, m)
  f = sum( (X - Z) ^ 2 )
}
```

Se recalculan las matrices de componentes mientras no se alcance el criterio de convergencia.

```
iter = 0
fold = f + (2 * conv * f)
while (fold - f > f * conv)
{
  iter = iter + 1
  fold = f

  Z = permnew(X, n, m, p)
  Z = permnew(t(B) %*% Z, r2, p, n)
  Z = permnew(t(C) %*% Z, r3, n, r2)
  A = qr.Q(qr(Z %*% (t(Z) %*% A)), complete = FALSE)

  Z = permnew(X, n, m, p)
  Z = permnew(Z, m, p, n)
  Z = permnew(t(C) %*% Z, r3, n, m)
  Z = permnew(t(A) %*% Z, r1, m, r3)
  B = qr.Q(qr(Z %*% (t(Z) %*% B)), complete = FALSE)

  Z = permnew(t(A) %*% X, r1, m, p)
```

```
    Z = permnew(t(B) %*% Z, r2, p, r1)
    C = qr.Q(qr(Z %*% (t(Z) %*% C)), complete = FALSE)

    Z = permnew(t(A) %*% X, r1, m, p)
    Z = permnew(t(B) %*% Z, r2, p, r1)
    H = permnew(t(C) %*% Z, r3, r1, r2)

    f = ss - sum( H ^ 2 )
  }
})

fp = 100 * (ss - f) / ss

La = H %*% t(H)
Y = permnew(H, r1, r2, r3)
Lb = Y %*% t(Y)
Y = permnew(Y, r2, r3, r1)
Lc = Y %*% t(Y)
```

Genera la salida.

```
out = list()
out$A = A
out$B = B
out$C = C
out$H = H
out$f = f
out$fp = fp
out$iter = iter
out$cputime = cputime[1]
out$La = La
out$Lb = Lb
```

```
    out$Lc = Lc
    return(out)
  }
  else
  {
    out = list()
  }
}
```

El procedimiento **T3PCAsegmented** aplica en primer lugar un modelo Tucker3 sobre el array de tres vías y a continuación un PCA sobre la matriz de dos vías, con la restricción de que la componente común sea igual para los dos modelos o bien, aplica primero un PCA sobre la matriz de dos vías y a continuación un Tucker3 sobre el array de tres vías con la restricción de que la componente común sea igual para los dos modelos. Hará una u otra cosa en función del parámetro *SegmentedType*. Si vale 1 se hará primero Tucker3 y después un constrained PCA, si vale 2 primero un PCA y después un contrained Tucker3.

En la Figura 7.3 se muestra un organigrama del procedimiento.

```
T3PCAsegmented <- function(X3D, Y,r1,r2, r3, conv, OriginalAlfa,
AlternativeLossF, nRuns, SegmentedType, StartSeed) {
    # INPUT X3D (n x m x p): 3D-data matrix (represented as a 3D-array)
    # [ X is concatenated version of X3D ]
    # Y (n x q): 2D-data matrix
    # r1,r2,r3: rank ofthe model Tucker3-model
    # conv: value for convergence (tolerance value)
    # OriginalAlfa (0-1): weight for the 3D-block
    # AlternativeLossF: using the alternative loss function?
    # nRuns: number of runs
    # SegmentedType: type of segmentation 1 = firstTucker,
    # next constrained PCA (common component matrix should be the same)
```

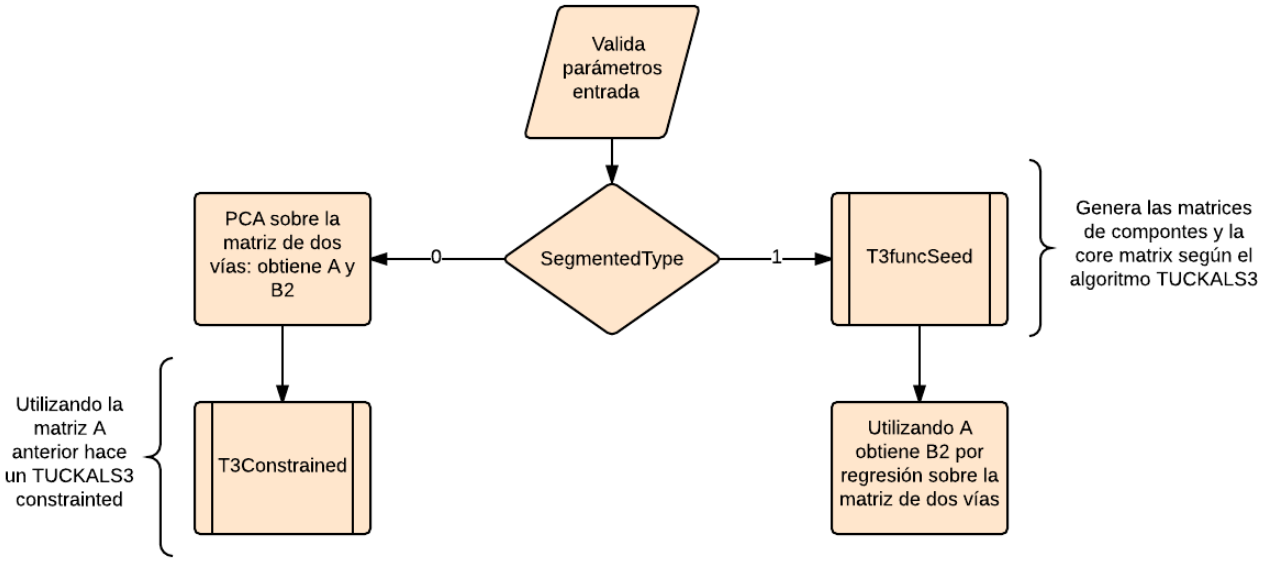


Figura 7.3: Organigrama del procedimiento **T3PCAsegmented**.

```
# 2 = first PCA, next constrained Tucker3 (common component
# matrix should be the same)
# StartSeed: seed to start the analysis
# OUTPUT Sol: structure forthe solution containing the
# following information - Info * AnalysisType * nRows * nColumns3D
# * nSlices * nColumns2D * RankVector * TolPercentage *
# OriginalAlfa * Alfa * AlternativeLossF * nRuns * SegmentedType
# * StartSeed - A - B1 - C - H - B2 - LossWeighted - LossUnweighted
# - FitPercentage -FitPercentage3D - nIter - FitValues - nIterValues
#  - La - Lb - Lc - Fit3D -Fit2D - Fit - Fit3Dsize - Fit2Dsize
# - Fitsize - CpuTime - TimeSeconds

set.seed(StartSeed)

source("T3constrained.R")
source("T3funcSeed.R")
```

```
source("GenerateSeed.R")

library(ThreeWay)

checkinput = 1

if ((SegmentedType != 1) & (SegmentedType != 2)) {
    cat(" ", fill = TRUE)
    cat("SegmentedType should be 1 (T3 first) or 2 (PCA first)"
    , fill = TRUE)
    cat(" ", fill = TRUE)
    checkinput = 0
}

if ((AlternativeLossF != 0) & (AlternativeLossF != 1)) {
    cat(" ", fill = TRUE)
    cat("AlternativeLossF should be 0 (original) or 1
    (alternative loss function)",fill = TRUE)
    cat(" ", fill = TRUE)
    checkinput = 0
}

if (checkinput == 1) {
    AnalysisSeeds = GenerateSeed(nRuns + 1, StartSeed)

    temp = dim(X3D)
    i = temp[1]
    j = temp[2]
    k = temp[3]
    rm(temp)

    temp = dim(Y)
```

```
l = temp[2]
rm(temp)

X = matrix(X3D, i, j * k)  # P slices are concatenated horizontally
ssq3D = sum(X^2)
ssq2D = sum(Y^2)

if (AlternativeLossF == 1) {
    Alfa = (OriginalAlfa * ssq2D)/((OriginalAlfa * ssq2D)
    + ((1 - OriginalAlfa) *ssq3D))
} else {
    Alfa = OriginalAlfa
}

FitValues = matrix(0, 1, nRuns + 1)
nIterValues = matrix(0, 1, nRuns + 1)

cputime = system.time({
    if (SegmentedType == 1) {

        # first Tucker3

        BestSol3D = T3funcSeed(X, i, j, k, r1, r2, r3, 0, 1e-07
        , AnalysisSeeds[1])
        FitValues[1] = BestSol3D$f
        nIterValues[1] = BestSol3D$iter
        for (runtel in 1:nRuns) {
          tempSol3D = T3funcSeed(X, i, j, k, r1, r2, r3, 1, 1e-07
          , AnalysisSeeds[runtel +1])
          FitValues[runtel + 1] = tempSol3D$f
          nIterValues[runtel + 1] = tempSol3D$iter
          if (tempSol3D$f < BestSol3D$f) {
```

```
      BestSol3D = tempSol3D
    }
    rm(tempSol3D)
  }
  BestA = BestSol3D$A  #orthonormal
  BestB1 = BestSol3D$B  #orthonormal
  BestC = BestSol3D$C  #orthonormal
  BestH = BestSol3D$H
  BestB2 = t(solve(t(BestA) %*% BestA) %*% t(BestA) %*% Y)
  #is in general not orthogonal !!
  BestIter = BestSol3D$iter
  BestFitPercentage3D = BestSol3D$fp
  Model2D = BestA %*% t(BestB2)
  Model3D = BestA %*% BestH %*% kronecker(t(BestC), t(BestB1))
  Loss3D = sum((Model3D - X)^2)
  Loss2D = sum((Model2D - Y)^2)
  FitPercentage = ((Alfa * (sum(BestH^2)/ssq3D)) + ((1 - Alfa) *
    (sum(Model2D^2)/ssq2D))) * 100
} else {

  # first PCA

  tempSol2D = svd(Y, r1, r1)
  EigValues = tempSol2D$d
  BestA = tempSol2D$u  # orthonormal
  BestB2 = tempSol2D$v %*% diag(EigValues[1:r1])
  BestSol3D = T3constrained(X, BestA, i, j, k, r1, r2, r3, 0,
    1e-07, AnalysisSeeds[1])
  FitValues[1] = BestSol3D$f
  nIterValues[1] = BestSol3D$iter
  for (runtel in 1:nRuns) {
    tempSol3D = T3constrained(X, BestA, i, j, k, r1, r2, r3, 1,
```

```
            1e-07, AnalysisSeeds[runtel + 1])
          FitValues[runtel + 1] = tempSol3D$f
          nIterValues[runtel + 1] = tempSol3D$iter
          if (tempSol3D$f < BestSol3D$f) {
            BestSol3D = tempSol3D
          }
          rm(tempSol3D)
        }
        BestB1 = BestSol3D$B
        BestC = BestSol3D$C
        BestH = BestSol3D$H
        BestIter = BestSol3D$iter
        BestFitPercentage3D = BestSol3D$fp
        Model3D = BestA %*% BestH %*% kronecker(t(BestC), t(BestB1))
        Model2D = BestA %*% t(BestB2)
        Loss3D = sum((Model3D - X)^2)
        Loss2D = sum((Model2D - Y)^2)
        FitPercentage = ((Alfa * (sum(BestH^2)/ssq3D)) + ((1 - Alfa) *
          (sum(Model2D^2)/ssq2D))) * 100
        rm(tempSol2D, EigValues)
    }
})

# compute intrinsic eigenvalues

La = BestH %*% t(BestH)
J = permnew(BestH, r1, r2, r3)
Lb = J %*% t(J)
J = permnew(J, r2, r3, r1)
Lc = J %*% t(J)

Out = list()
```

```
Out$Info = list()
if (SegmentedType == 1) {
    Out$Info$AnalysisType = "segmented 3D"
} else {
    Out$Info$AnalysisType = "segmented 2D"
}
Out$Info$nRows = i
Out$Info$nColumns3D = j
Out$Info$nSlices = k
Out$Info$nColumns2D = l
Out$Info$RankVector = cbind(r1, r2, r3)
Out$Info$TolPercentage = conv
Out$Info$OriginalAlfa = OriginalAlfa
Out$Info$Alfa = Alfa
Out$Info$AlternativeLossF = AlternativeLossF
Out$Info$nRuns = nRuns
Out$Info$SegmentedType = SegmentedType
Out$Info$StartSeed = StartSeed
Out$A = BestA
Out$B1 = BestB1
Out$C = BestC
Out$H = BestH
Out$B2 = BestB2
Out$LossWeighted = (Alfa * Loss3D) + ((1 - Alfa) * Loss2D)
Out$LossUnweighted = Loss3D + Loss2D
Out$FitPercentage = FitPercentage
Out$FitPercentage3D = BestFitPercentage3D
Out$nIter = BestIter
Out$FitValues = FitValues
Out$nIterValues = nIterValues
Out$La = La
Out$Lb = Lb
```

```
        Out$Lc = Lc
        Out$Fit3D = Loss3D
        Out$Fit2D = Loss2D
        Out$Fit = (Alfa * Loss3D) + ((1 - Alfa) * Loss2D)
        Out$Fit3Dsize = Loss3D/(i * j * k)
        Out$Fit2Dsize = Loss2D/(i * l)
        Out$Fitsize = Out$Fit/((i * j * k) + (i * l))
        Out$CpuTime = cputime[1]
        Out$TimeSeconds = round(cputime[1], 2)
        return(Out)
    } else {
        Out = list()
    }
}
```

Para hacer un Constrained Tucker3 se utiliza la función **T3constrained**. Esta función ajusta un modelo Tucker3 con **B** y **C** ortonormales y la matriz **A** igual a una matriz A que recibe como parámetro (esta matriz también es ortonormal).

```
T3constrained <- function(X, FixedA, n, m, p, r1, r2, r3, start, conv = 1e-07
,StartSeed, StartB, StartC, StartH) {

    # fit a Tucker3 model (with orthonormal B and C) with the constraint
    # that A should equal the given BestA (ideally FixedA is also orthonormal)

    # X (i x jk): 3D-data matrix (displayed in matrix form)
    # FixedA (i x r1): fixed A component matrix (for the row mode)
    # [ideally A is orthonormal but is not necessary]
    # n: number of rows
    # m: number of columns
    # p: number of slices
    # r1: number of component for the row-mode A
    # r2: number of componentfor the column-mode B
```

```
# r3: number of component for the slice-mode C
# start: type of starting values for B and C 0 = generalized eigenvalue
# decomposition 1 = random starting point 2 = user specified component
# matrices conv: tolerance value (default: .0000001)
# StartSeed: seed tostart the analysis
# [ optional: specify StartB, StartC and StartH ]
# StartB (j x r2): [optional] initial values for the component matrix
# for the column-mode B (should be orthonormal)
# StartC (k x r3): [optional] initialvalues for the component matrix
# for the slice-mode C (should be orthonormal)
# StartH (r1 x r2r3): [optional] initial values for the core array H

set.seed(StartSeed)
library(ThreeWay)

checkinput = 1

if (sum(((t(FixedA) %*% FixedA) - diag(r1))^2) > 1e-07) {
    cat(" ", fill = TRUE)
    cat("A is not orthonormal (which is not necessary but recommended)",
        fill = TRUE)
    cat(" ", fill = TRUE)
}

if ((start != 0) & (start != 1) & (start != 2)) {
    cat(" ", fill = TRUE)
    cat("start should be 0 (rational start), 1 (random start)
    or 2 (user specified start)",
        fill = TRUE)
    cat(" ", fill = TRUE)
    checkinput = 0
}
```

```
if (checkinput == 1) {
    if (length(dim(X)) != 2) {
        X = matrix(X, n, m * p)  # P slices are concatenated horizontally
    }
    X = as.matrix(X)

    cputime = system.time({
        ss = sum(X^2)

        # determine initial B and C

        if (start == 0) {
            Z = permnew(X, n, m, p)
            EIG = eigen(Z %*% t(Z))
            B = EIG$vectors[, 1:r2]
            Z = permnew(Z, m, p, n)
            EIG = eigen(Z %*% t(Z))
            C = EIG$vectors[, 1:r3]
        } else {
            if (start == 1) {
              if (m >= r2) {
                B = orth(matrix(runif(m * r2, 0, 1), m, r2) - 0.5)
              } else {
                B = orth(matrix(runif(r2 * r2, 0, 1), r2, r2) - 0.5)
                B = B[1:m, ]
              }

              if (p >= r3) {
                C = orth(matrix(runif(p * r3, 0, 1), p, r3) - 0.5)
              } else {
                C = orth(matrix(runif(r3 * r3, 0, 1), r3, r3) - 0.5)
                C = C[1:p, ]
```

```
      }
    } else {
      if (sum(((t(StartB) %*% StartB) - diag(r2))^2) > 1e-07) {
        B = orth(StartB)
      } else {
        B = StartB
      }

      if (sum(((t(StartC) %*% StartC) - diag(r3))^2) > 1e-07) {
        C = orth(StartC)
      } else {
        C = StartC
      }

      H = StartH
    }
}

# compute loss (and eventually initial H)

if (start != 2) {
    Z = permnew(t(FixedA) %*% X, r1, m, p)
    Z = permnew(t(B) %*% Z, r2, p, r1)
    H = permnew(t(C) %*% Z, r3, r1, r2)
    f = ss - sum(H^2)
} else {
    Z = B %*% permnew(FixedA %*% H, n, r2, r3)
    Z = C %*% permnew(Z, m, r3, n)
    Z = permnew(Z, p, n, m)
    f = sum((X - Z)^2)
}
```

```
    iter = 0
    fold = f + (2 * conv * f)
    while ((fold - f) > (f * conv)) {
        iter = iter + 1
        fold = f

        Z = permnew(X, n, m, p)
        Z = permnew(Z, m, p, n)
        Z = permnew(t(C) %*% Z, r3, n, m)
        Z = permnew(t(FixedA) %*% Z, r1, m, r3)
        B = qr.Q(qr(Z %*% (t(Z) %*% B)), complete = FALSE)
        rm(Z)

        Z = permnew(t(FixedA) %*% X, r1, m, p)
        Z = permnew(t(B) %*% Z, r2, p, r1)
        C = qr.Q(qr(Z %*% (t(Z) %*% C)), complete = FALSE)
        rm(Z)

        Z = permnew(t(FixedA) %*% X, r1, m, p)
        Z = permnew(t(B) %*% Z, r2, p, r1)
        H = permnew(t(C) %*% Z, r3, r1, r2)
        rm(Z)

        f = ss - sum(H^2)
    }
})

fp = 100 * ((ss - f)/ss)

Out = list()
Out$B = B
Out$C = C
```

```
        Out$H = H
        Out$f = f
        Out$fp = fp
        Out$iter = iter
        Out$cputime = cputime[1]
        return(Out)
    }
}
```

7.2.3. ESTRATEGIA SEPARADA: T3PCAseparate.R

El procedimiento **T3PCAseparate** aplica de forma independiente la descomposición Tuckals3 sobre el array de tres vías y la descomposición PCA sobre la matriz de dos vías

En la Figura 7.4 se muestra un organigrama del procedimiento.

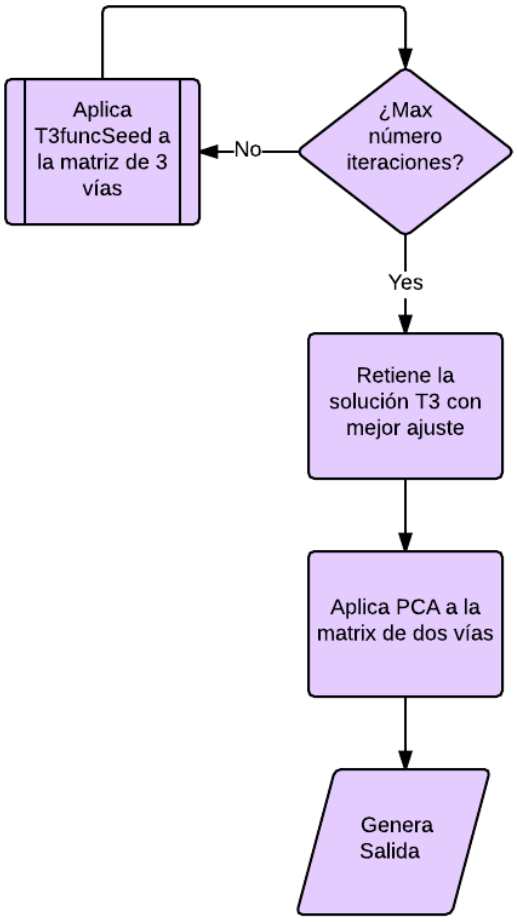


Figura 7.4: Organigrama del procedimiento T3PCAseparate.

```
T3PCAseparate <- function(X3D, Y, r1, r2, r3, conv, OriginalAlfa, AlternativeLossF,
    nRuns, StartSeed) {
    # estimates a T3 for X3D (with orthonormal A, B1 and C) and a PCA for Y
    # (orthonormal A and orthogonal B2)

    # INPUT X3D (n x m x p): 3D-data array
    # Y (n x q): 2D-data matrix
    # r1,r2,r3: rank of the model Tucker3-model
    # conv: value for convergence (tolerance value)
    # OriginalAlfa (0-1): weight for the 3D-block
    # AlternativeLossF: using the alternative loss function?
    # nRuns: number of runs
    # StartSeed: seed to start the analysis
    # OUTPUT Sol:structure for the solution containing the following information:
    # Info * AnalysisType * nRows * nColumns3D * nSlices * nColumns2D *
    # RankVector * TolPercentage * OriginalAlfa * Alfa * AlternativeLossF *
    # nRuns * StartSeed - A1 - B1 - C - H - A2 - B2 - LossWeighted -
    # LossUnweighted - FitPercentage - FitPercentage3D - nIter -
    # FitValues - nIterValues - La - Lb - Lc - Fit3D - Fit2D - Fit -
    # Fit3Dsize - Fit2Dsize - Fitsize - CpuTime- TimeSeconds

    set.seed(StartSeed)
    library(ThreeWay)
    source("T3funcSeed.R")

    AnalysisSeeds = GenerateSeed(nRuns + 1, StartSeed)

    temp = dim(X3D)
    i = temp[1]
    j = temp[2]
    k = temp[3]
    rm(temp)
```

```
temp = dim(Y)
l = temp[2]
rm(temp)

X = matrix(X3D, i, j * k)  # P slices are concatenated horizontally
ssq3D = sum(X^2)
ssq2D = sum(Y^2)

if (AlternativeLossF == 1) {
    Alfa = (OriginalAlfa * ssq2D)/((OriginalAlfa * ssq2D) +
    ((1 - OriginalAlfa) *ssq3D))
} else {
    Alfa = OriginalAlfa
}

FitValues = matrix(0, 1, nRuns + 1)
nIterValues = matrix(0, 1, nRuns + 1)

cputime = system.time({

    # Tucker3 on X

    BestSol3D = T3funcSeed(X, i, j, k, r1, r2, r3, 0, 1e-07
    , AnalysisSeeds[1])
    FitValues[1] = BestSol3D$f
    nIterValues[1] = BestSol3D$iter
    for (runtel in 1:nRuns) {
        tempSol3D = T3funcSeed(X, i, j, k, r1, r2, r3, 1, 1e-07
        , AnalysisSeeds[runtel +1])
        FitValues[runtel + 1] = tempSol3D$f
        nIterValues[runtel + 1] = tempSol3D$iter
        if (tempSol3D$f < BestSol3D$f) {
```

```
            BestSol3D = tempSol3D
        }
        rm(tempSol3D)
    }
    BestA1 = BestSol3D$A  #orthonormal
    BestB1 = BestSol3D$B  #orthonormal
    BestC = BestSol3D$C  #orthonorma
    BestH = BestSol3D$H
    BestIter = BestSol3D$iter
    BestFitPercentage3D = BestSol3D$fp

    # PCA on Y

    tempSol2D = svd(Y, r1, r1)
    EigValues = tempSol2D$d
    BestA2 = tempSol2D$u  #orthonormal
    BestB2 = tempSol2D$v %*% diag(EigValues[1:r1])
    rm(tempSol2D, EigValues)
    Model3D = BestA1 %*% BestH %*% kronecker(t(BestC), t(BestB1))
    Model2D = BestA2 %*% t(BestB2)
    Loss3D = sum((Model3D - X)^2)
    Loss2D = sum((Model2D - Y)^2)
    FitPercentage = ((Alfa * (sum(BestH^2)/ssq3D)) + ((1 - Alfa) *
    (sum(Model2D^2)/ssq2D))) *
        100
})

# compute intrinsic eigenvalues

La = BestH %*% t(BestH)
J = permnew(BestH, r1, r2, r3)
Lb = J %*% t(J)
```

```
J = permnew(J, r2, r3, r1)
Lc = J %*% t(J)

Out = list()
Out$Info = list()
Out$Info$AnalysisType = "separate"
Out$Info$nRows = i
Out$Info$nColumns3D = j
Out$Info$nSlices = k
Out$Info$nColumns2D = l
Out$Info$RankVector = cbind(r1, r2, r3)
Out$Info$TolPercentage = conv
Out$Info$OriginalAlfa = OriginalAlfa
Out$Info$Alfa = Alfa
Out$Info$AlternativeLossF = AlternativeLossF
Out$Info$nRuns = nRuns
Out$Info$StartSeed = StartSeed
Out$A1 = BestA1
Out$B1 = BestB1
Out$C = BestC
Out$H = BestH
Out$A2 = BestA2
Out$B2 = BestB2
Out$LossWeighted = (Alfa * Loss3D) + ((1 - Alfa) * Loss2D)
Out$LossUnweighted = Loss3D + Loss2D
Out$FitPercentage = FitPercentage
Out$FitPercentage3D = BestFitPercentage3D
Out$nIter = BestIter
Out$FitValues = FitValues
Out$nIterValues = nIterValues
Out$La = La
Out$Lb = Lb
```

```
    Out$Lc = Lc
    Out$Fit3D = Loss3D
    Out$Fit2D = Loss2D
    Out$Fit = (Alfa * Loss3D) + ((1 - Alfa) * Loss2D)
    Out$Fit3Dsize = Loss3D/(i * j * k)
    Out$Fit2Dsize = Loss2D/(i * l)
    Out$Fitsize = Out$Fit/((i * j * k) + (i * l))
    Out$CpuTime = cputime[1]
    Out$TimeSeconds = round(cputime[1], 2)
    return(Out)
}
```

7.3. GENERA DATOS: GenerateTucker3-PCA.R

El procedimiento **GenerateTucker3-PCA.R** a partir de los parámetros que recibe genera de forma aleatoria las matrices de componentes $\mathbf{A}$, $\mathbf{B}_1$, $\mathbf{C}$ y $\mathbf{B}_2$ (además genera las matrices $\mathbf{A}_1$ y $\mathbf{A}_2$ cuando se trabaja con una componente adicional) y la core matrix $\underline{\mathbf{G}}$. A partir de estas construye el modelo real (los arrays de tres y dos vías), a los que les añade el ruido.

En la Figura 7.5 se muestra un organigrama del procedimiento.

```
GenerateTucker3_PCA <- function(i, j, k, l, rank1, rank2, rank3, noise
, ExplVarVector, OriginalAlfa, AlternativeLossF,nExtraComponents,
 GenerateSeed) {

    # generates a Tucker3_PCA true model and data

    # i: number of elements of first mode (the common mode: rows)
    # j: number of elements of second mode (columns)
    # k: number of elements of third mode (slabs)
    # l: number of covariates
    # rank1,rank2,rank3: rank of the model
    # noise (0-1): amount of noise added to the model
    # ExplVarVector(1 x rank1 + 1): elements denote percentage of
    # explained variance per component for the COMMON mode
    # (should sum to one)
    # OriginalAlfa: weight for the 3D-block (1-alfa is weight for
    # the 2D-block
    # AlternativeLossF: using the alternative loss function?
    # 0 = no (use original loss function: weighted SSQ)
    # 1 = yes (useweighted loss function with scaled SSQ:
    # scaled by the SSQ in X and Y)
```

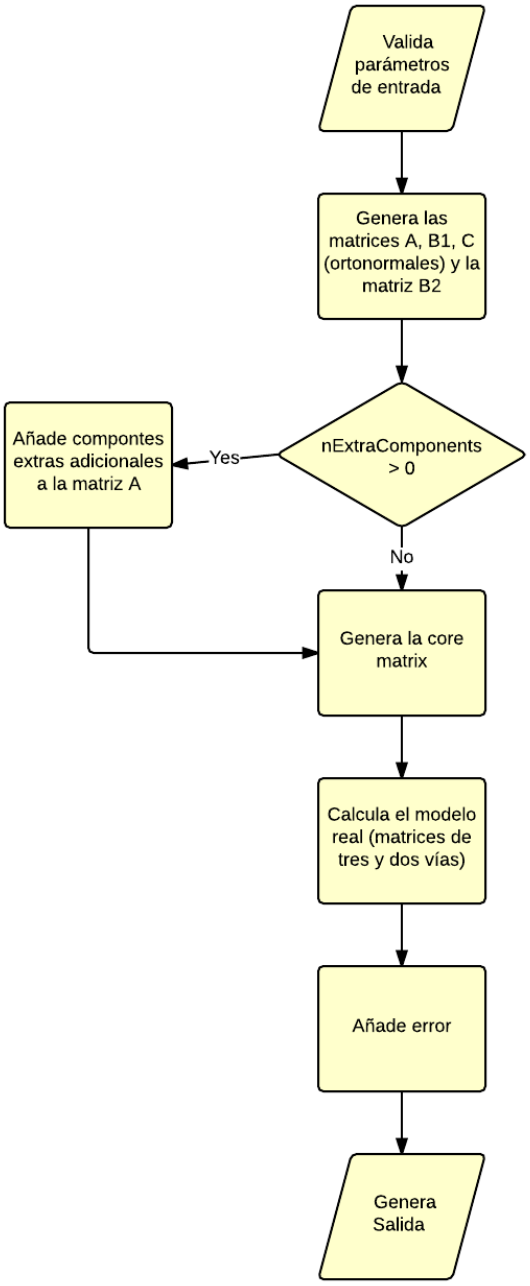


Figura 7.5: Organigrama del procedimiento $GenerateTucker3 - PCA$.

```
# nExtraComponents: number of extra components for the common mode
# GenerateSeed: seed for the generation
# OUTPUT Sol: structure for the solution containing the following
# information- nRow - nCol1 - nSlice - nCol2 - RankVector - Error
# - ExplVarVector - OriginalAlfa - AlternativeLossF
# - nExtraComponents - GenerateSeed - A - B1 - C - B2orth - B2
# - A1 - A2 - Core - CoreMatrix - TrueWay3DMatrix
# - TrueWay3DMatrixNoExtraComp - TrueWay3D - TrueWay3DnoExtraComp
# - TrueWay2D - TrueWay2DnoExtraComp - Way3D - Way3DMatrix - Way2D
# - BOD3D - BOD2D - BOD3Dsize - BOD2Dsize - BOD - BODsize

library(ThreeWay, MASS)

set.seed(GenerateSeed)  # rnorm and runif used

# check input

checkinput = 1

if (rank1 > min(i, j * k, l)) {
    cat(" ", fill = TRUE)
    cat("rank1 should be an integer between 1 and ", min(i, l, j * k),
    fill = TRUE)
    cat(" ", fill = TRUE)
    checkinput = 0
}

if (rank2 > min(j, i * k)) {
    cat(" ", fill = TRUE)
    cat("rank2 should be an integer between 1 and ", min(j, i * k),
    fill = TRUE)
    cat(" ", fill = TRUE)
```

```
    checkinput = 0
}

if (rank3 > min(k, i * j)) {
    cat(" ", fill = TRUE)
    cat("rank3 should be an integer between 1 and ", min(k, i * j),
    fill = TRUE)
    cat(" ", fill = TRUE)
    checkinput = 0
}

if ((rank1 > rank2 * rank3) | (rank2 > rank1 * rank3) | (rank3 > rank1
 * rank2)) {
    cat(" ", fill = TRUE)
    cat("None of the ranks can be larger than the products of the
     other two (e.g., rank1 > rank2*rank3 is not allowed",
        fill = TRUE)
    cat(" ", fill = TRUE)
    checkinput = 0
}

if ((noise < 0) || (noise >= 1)) {
    cat(" ", fill = TRUE)
    cat("Noise is wrong. Should be a value between 0 and 1
    (0 is possible, 1 not)", fill = TRUE)
    cat(" ", fill = TRUE)
    checkinput = 0
}

if (length(ExplVarVector) != (rank1 + nExtraComponents)) {
    cat(" ", fill = TRUE)
    cat("ExplVarVector should contain ", rank1 + 1, " elements",
```

```
        fill = TRUE)
        cat(" ", fill = TRUE)
        checkinput = 0
    } else {
        if (abs(sum(ExplVarVector) - 1) > 1e-10) {
            cat(" ", fill = TRUE)
            cat("ExplVarVector should sum to one", fill = TRUE)
            cat(" ", fill = TRUE)
            checkinput = 0
        } else {
            if (sum(sign(ExplVarVector) == -1) > 0) {
                cat(" ", fill = TRUE)
                cat("ExplVarVector should only contain positive values
                (between 0 and 1)", fill = TRUE)
                cat(" ", fill = TRUE)
                checkinput = 0
            }
        }
    }

    if ((nExtraComponents < 0) | ((nExtraComponents%%1) != 0)) {
        cat(" ", fill = TRUE)
        cat("nExtraComponents should be a positive integer (or zero)",
        fill = TRUE)
        cat(" ", fill = TRUE)
        checkinput = 0
    }

    if ((OriginalAlfa < 0) || (OriginalAlfa > 1)) {
        cat(" ", fill = TRUE)
        cat("OriginalAlfa should be between 0 and 1", fill = TRUE)
        cat(" ", fill = TRUE)
```

```
    checkinput = 0
}

if ((AlternativeLossF != 0) && (AlternativeLossF != 1)) {
    cat(" ", fill = TRUE)
    cat("AlternativeLossF should be 0 or 1", fill = TRUE)
    cat(" ", fill = TRUE)
    checkinput = 0
}

if (checkinput == 1) {

    # generate matrices

    Sol = list()
    Sol$nRow = i
    Sol$nCol1 = j
    Sol$nSlice = k
    Sol$nCol2 = l
    Sol$RankVector = cbind(rank1, rank2, rank3)
    Sol$Error = noise
    Sol$ExplVarVector = ExplVarVector
    Sol$OriginalAlfa = OriginalAlfa
    Sol$AlternativeLossF = AlternativeLossF
    Sol$nExtraComponents = nExtraComponents
    Sol$GenerateSeed = GenerateSeed

    # centered component matrices (A, B1 and C orthonormal; B2 is
    # taken orthogonal)

    Sol$A = orth(scale(matrix(rnorm(i * (rank1 + (2 * nExtraComponents)),
        0, 1), i, rank1 + (2 * nExtraComponents)), T, F))
```

```
Sol$B1 = orth(scale(matrix(rnorm(j * rank2, 0, 1), j, rank2), T, F))
Sol$C = orth(scale(matrix(rnorm(k * rank3, 0, 1), k, rank3), T, F))
Sol$B2orth = orth(scale(matrix(rnorm(l * (rank1 + nExtraComponents),
    0, 1), l, rank1 + nExtraComponents), T, F))
Sol$B2 = matrix(0, l, rank1 + nExtraComponents)

if (nExtraComponents > 0) {
    Sol$A1 = Sol$A[, cbind(t(1:rank1), t((rank1 + 1):(rank1 +
    nExtraComponents)))]
    Sol$A2 = Sol$A[, cbind(t(1:rank1), t((rank1 + 1 +
    nExtraComponents):(rank1 +
        nExtraComponents + nExtraComponents)))]

    # Sol£A2 = Sol£A1 ## in case you only want common components

} else {
    Sol$A1 = Sol$A
    Sol$A2 = Sol$A
}

# generate core structure (adapted to incorporate additional
#component)

Sol$Core = array(rnorm((rank1 + nExtraComponents) * rank2 * rank3,
0, 1), cbind(rank1 + nExtraComponents, rank2, rank3))
CoreSq = Sol$Core * Sol$Core
TotalSqCore = sum(CoreSq)

for (rowtel in 1:dim(Sol$Core)[1]) {
    tempweight = (ExplVarVector[rowtel]/sum(CoreSq[rowtel, , ])) *
    TotalSqCore
    Sol$Core[rowtel, , ] = Sol$Core[rowtel, , ] * sqrt(tempweight)
```

```
    rm(tempweight)
}

# convert three-way core array (q1 x q2 x q3) to a core matrix
# (q1 x q2q3)

Sol$CoreMatrix = matrix(Sol$Core, rank1 + nExtraComponents, rank2 *
    rank3)

# Computing the true model

Sol$TrueWay3DMatrix = matrix(0, i, j * k)
Sol$TrueWay3DMatrixNoExtraComp = matrix(0, i, j * k)
for (rowtel in 1:(rank1 + nExtraComponents)) {
    for (coltel in 1:rank2) {
        for (slicetel in 1:rank3) {
          temparray = Sol$Core[rowtel, coltel, slicetel] * Sol$A1[,
            rowtel] %*% kronecker(t(Sol$C[, slicetel]),
            t(Sol$B1[, coltel]))  # is i-jk
          Sol$TrueWay3DMatrix = Sol$TrueWay3DMatrix + temparray
          if (rowtel <= rank1) {
            Sol$TrueWay3DMatrixNoExtraComp
            = Sol$TrueWay3DMatrixNoExtraComp
            + temparray
          }
          rm(temparray)
        }
    }
}

# convert matricized data matrix to a 3D-data array
```

```
Sol$TrueWay3D = array(Sol$TrueWay3DMatrix, cbind(i, j, k))
Sol$TrueWay3DnoExtraComp = array(Sol$TrueWay3DMatrixNoExtraComp,
cbind(i, j, k))

EigVals2D = runif(rank1 + nExtraComponents, 0, 1)
TotalArraySq = sum(EigVals2D * EigVals2D)
Sol$TrueWay2D = matrix(0, i, l)
Sol$TrueWay2DnoExtraComp = matrix(0, i, l)
for (rowtel in 1:(rank1 + nExtraComponents)) {
    tempweight = ExplVarVector[rowtel] * TotalArraySq
    tempmatrix = sqrt(tempweight) * Sol$A2[, rowtel] %*%
    t(Sol$B2orth[,
        rowtel])
    Sol$TrueWay2D = Sol$TrueWay2D
    + tempmatrix
    if (rowtel <= rank1) {
        Sol$TrueWay2DnoExtraComp = Sol$TrueWay2DnoExtraComp + tempmatrix
    }
    rm(tempmatrix)
    rm(tempweight)
}

EigVal = sqrt(ExplVarVector * TotalArraySq)
tempIdentity = diag(rank1 + nExtraComponents)
diag(tempIdentity) = EigVal
Sol$B2 = Sol$B2orth %*% tempIdentity
Sol$EigenValues2D = EigVal
rm(EigVal, tempIdentity)

# adding error

error3D = array(rnorm(i * j * k, 0, 1), cbind(i, j, k))
```

```
error2D = matrix(rnorm(i * l, 0, 1), i, l)
error3D = error3D * sqrt(sum(Sol$TrueWay3D^2)/sum(error3D^2))
error2D = error2D * sqrt(sum(Sol$TrueWay2D^2)/sum(error2D^2))
errorlevel = noise/(1 - noise)
Sol$Way3D = Sol$TrueWay3D + (error3D * sqrt(errorlevel))
Sol$Way3DMatrix = matrix(Sol$Way3D, i, j * k)
Sol$Way2D = Sol$TrueWay2D + (error2D * sqrt(errorlevel))

# computing badness-of-data

if (AlternativeLossF == 1) {
    ssq3D = sum(Sol$Way3D^2)
    ssq2D = sum(Sol$Way2D^2)
    Alfa = (OriginalAlfa * ssq2D)/((OriginalAlfa * ssq2D) +
    ((1 - OriginalAlfa) * ssq3D))
} else {
    Alfa = OriginalAlfa
}

Sol$BOD3D = sum((Sol$Way3D - Sol$TrueWay3DnoExtraComp)^2)
Sol$BOD2D = sum((Sol$Way2D - Sol$TrueWay2DnoExtraComp)^2)
Sol$BOD3Dsize = Sol$BOD3D/(i * j * k)
Sol$BOD2Dsize = Sol$BOD2D/(i * l)
Sol$BOD = (Alfa * Sol$BOD3D) + ((1 - Alfa) * Sol$BOD2D)
Sol$BODsize = Sol$BOD/((i * j * k) + (i * l))

Sol$BOD3DExtra = sum((Sol$Way3D - Sol$TrueWay3D)^2)
Sol$BOD2DExtra = sum((Sol$Way2D - Sol$TrueWay2D)^2)
Sol$BOD3DsizeExtra = Sol$BOD3DExtra/(i * j * k)
Sol$BOD2DsizeExtra = Sol$BOD2DExtra/(i * l)
Sol$BODExtra = (Alfa * Sol$BOD3DExtra) + ((1 - Alfa) * Sol$BOD2DExtra)
Sol$BODsizeExtra = Sol$BODExtra/((i * j * k) + (i * l))
```

```
        return(Sol)
    }
}
```

7.4. CALCULA RECOVERY

Para calcular el Recovery se utilizan dos funciones **ComputeRecoveryT3PCA.R** y **ComputeRecoveryT3PCAExtra.R**. Ambas funciones utilizan a su vez las funciones siguientes:

- **procr.R**: Encuentra la rotación ortogonal de una matriz (primer parámetro) que mejor ajusta a otra (segundo parámetro) en el sentido de los mínimos cuadrados alternados.

```
procr <- function(X, Ytarget) {
    # find orthonormal rotation of X that fits Ytarget best
    # in least squares sense

    # INPUT X (n x p1): matrix that will be rotated (orthogonally)
    #  Ytarget (n x p2): target matrix

    # if p1 >= p2: Tmat is orthonormal
    # if p1 < p2: t(Tmat) is orthonormal (Tmat is not orthonormal)

    if (dim(X)[1] == dim(Ytarget)[1]) {
        source("ComputePhiMatrix.R")
        temp = svd(t(X)
        %*% Ytarget)
        Tmat = temp$u %*% t(temp$v)
        Yhat = X %*% Tmat  # (n x p2)
        RotationMatrix = Tmat
        Fit = sum((Yhat - Ytarget)^2)
        PhiMatrix = ComputePhiMatrix(Ytarget, Yhat)

        RotationMatrixNonorthogonal = ginv(t(X) %*% X) %*% t(X)
        %*% Ytarget
```

```
        YhatNonorthogonal = X %*% RotationMatrixNonorthogonal
        FitNonorthogonal = sum((YhatNonorthogonal - Ytarget)^2)
        PhiMatrixNonorthogonal = ComputePhiMatrix
        (Ytarget, YhatNonorthogonal)

        Out = list()
        Out$Yhat = Yhat  #orthogonally rotated X matrix
        Out$RotationMatrix = RotationMatrix
        Out$Fit = Fit
        Out$PhiMatrix = PhiMatrix
        Out$TuckerCongruence = mean(abs(diag(PhiMatrix)))

        Out$NonOrthogonal$Yhat = YhatNonorthogonal
        Out$NonOrthogonal$RotationMatrix = RotationMatrixNonorthogonal
        Out$NonOrthogonal$Fit = FitNonorthogonal
        Out$NonOrthogonal$PhiMatrix = PhiMatrixNonorthogonal
        Out$NonOrthogonal$TuckerCongruence =
        mean(abs(diag(PhiMatrixNonorthogonal)))
    } else {
        cat(" ", fill = TRUE)
        cat("X and Ytarget should have the same number of rows",
        fill = TRUE)
        cat(" ", fill = TRUE)
        Out = list()
    }
    return(Out)
}
```

Esta función a su vez utiliza la función **ComputePhiMatrix.R** que calcula que coeficiente Phi de Tucker.

```
ComputePhiMatrix <- function ( Ytarget , Yhat )

# computes Tucker's phi among columns of input matrix

{
  Out = solve( diag( diag( t(Ytarget)
  %*% Ytarget ) ) ^ .5 ) %*% t(Ytarget) %*% Yhat %*%
  solve( diag( diag( t(Yhat) %*% Yhat ) ) ^ .5 )
  return( Out )
}
```

- **ComputeTuckerMatrix.R** que calcula la congruenica (Tucker) entre dos matrices

```
ComputeTuckerMatrix <- function(X, Y)

# Computes the Tucker congruence between two matrices

# X (n x p): first matrix Y (n x p): second matrix

{
    source("ComputePhiMatrix.R")
    PhiMatrix = ComputePhiMatrix(X, Y)
    Tucker = mean(abs(diag(PhiMatrix)))
    return(Tucker)
}
```

7.4.1. ComputeRecoveryT3PCA.R

ComputeRecoveryT3PCA.R calcula las congruencias entre la solución real (sin componente extra adicional) y la solución T3-PCA obtenida por el algoritmo.

```
ComputeRecoveryT3PCA <- function(TrueSol, Sol, SeparateAnalysis, ProxyFit)
{
    # compute congruences between true (without extra component)
    #  and obtained T3PCA-solution

    # SeparateAnalysis: Sol obtained from the separate analysis strategy?
    # 0 = no (Sol from simultaneous or segmented strategy)
    # 1 = yes (Sol from separate analysis)
    # Extracomp: solution with extra component 0 = no (no extra component)
    # 1 = yes

    library(MASS)
    source("procr.R")
    source("ComputeTuckerMatrix.R")

    checkinput = 1

    if ((SeparateAnalysis != 0) & (SeparateAnalysis != 1)) {
        cat(" ", fill = TRUE)
        cat("SeparateAnalysis should be 0 (simultaneous/segmented)
        or 1 (separate)",
            fill = TRUE)
        cat(" ", fill = TRUE)
        checkinput = 0
    }

    Out = list()

    if (checkinput == 1) {

        # Comparing badness-of-fit and badness-of-data
```

```
Out$BOFBOD = Sol$Fit - TrueSol$BOD
Out$BOFBOD3D = Sol$Fit3D - TrueSol$BOD3D
Out$BOFBOD2D = Sol$Fit2D - TrueSol$BOD2D
Out$BOFBODsize = Sol$Fitsize - TrueSol$BODsize
Out$BOFBOD3Dsize = Sol$Fit3Dsize - TrueSol$BOD3Dsize
Out$BOFBOD2Dsize = Sol$Fit2Dsize - TrueSol$BOD2Dsize

Out$BODreached = Sol$Fit <= ProxyFit
Out$BODdiff = Sol$Fit - ProxyFit
Out$nRunsOptimalFit = sum(Sol$FitValues == min(Sol$FitValues))

# Recovery of M (BOR)

r1 = Sol$Info$RankVector[1]
r2 = Sol$Info$RankVector[2]
r3 = Sol$Info$RankVector[3]

if (SeparateAnalysis == 1) {
    Model3D = Sol$A1 %*% matrix(Sol$H, r1, r2 * r3) %*%
    kronecker(t(Sol$C),t(Sol$B1))  Model2D = Sol$A2 %*% t(Sol$B2)
} else {
    Model3D = Sol$A %*% matrix(Sol$H, r1, r2 * r3) %*%
    kronecker(t(Sol$C),t(Sol$B1))Model2D = Sol$A %*% t(Sol$B2)
}

Out$Recov = 0
Out$Recov3D = sum((Model3D - TrueSol$TrueWay3DMatrixNoExtraComp)^2)
Out$Recov2D = sum((Model2D - TrueSol$TrueWay2DnoExtraComp)^2)
Out$Recov = (Sol$Info$Alfa * Out$Recov3D) + ((1 -Sol$Info$Alfa)
* Out$Recov2D)
Out$Recovsize = Out$Recov/((Sol$Info$nRows * Sol$Info$nColumns3D *
Sol$Info$nSlices) +(Sol$Info$nRows *  Sol$Info$nColumns2D))
```

```
Out$Recov3Dsize = Out$Recov3D/(Sol$Info$nRows * Sol$Info$nColumns3D *
    Sol$Info$nSlices)
Out$Recov2Dsize = Out$Recov2D/(Sol$Info$nRows * Sol$Info$nColumns2D)

Out$RecovExtra = 0
Out$Recov3DExtra = sum((Model3D -
TrueSol$TrueWay3DMatrix)^2)
Out$Recov2DExtra = sum((Model2D - TrueSol$TrueWay2D)^2)
Out$RecovExtra = (Sol$Info$Alfa * Out$Recov3DExtra)
+ ((1 - Sol$Info$Alfa) *Out$Recov2DExtra)
Out$RecovsizeExtra = Out$RecovExtra/((Sol$Info$nRows
* Sol$Info$nColumns3D *Sol$Info$nSlices)
+ (Sol$Info$nRows * Sol$Info$nColumns2D))
Out$Recov3DsizeExtra = Out$Recov3DExtra/
(Sol$Info$nRows *Sol$Info$nColumns3D *Sol$Info$nSlices)
Out$Recov2DsizeExtra = Out$Recov2DExtra/
(Sol$Info$nRows *Sol$Info$nColumns2D)

if (SeparateAnalysis == 1) {

    # Recovery measures for component matrices

    optrotationA1 = procr(Sol$A1, TrueSol$A[, 1:r1])
    optrotationB1 = procr(Sol$B1, TrueSol$B1)
    optrotationC = procr(Sol$C, TrueSol$C)

    # counterrotation for the core

    # (based on optimal rotations for A, B and C)

    InvRotatA1 = ginv(optrotationA1$RotationMatrix)
    rotatedCore = InvRotatA1 %*% Sol$H %*%
```

```
    kronecker(t(ginv(optrotationC$RotationMatrix)),
    t(ginv(optrotationB1$RotationMatrix)))

    # counterrotation for E (based on optimal rotation for A)

    optrotationA2 = procr(Sol$A2, TrueSol$A[, 1:r1])
    InvRotatA2 = ginv(optrotationA2$RotationMatrix)
    rotationB2 = t(InvRotatA2 %*% t(Sol$B2))
    optrotationB2 = procr(Sol$B2, TrueSol$B2[, 1:r1])
    InvRotatB2 = ginv(optrotationB2$RotationMatrix)
    rotationA2 = Sol$A2 %*% InvRotatB2
} else {

    # Recovery measures for component matrices

    optrotationA = procr(Sol$A, TrueSol$A[, 1:r1])
    optrotationB1 = procr(Sol$B1, TrueSol$B1)
    optrotationC = procr(Sol$C, TrueSol$C)

    # counterrotation for the core

    # (based on optimal rotations for A, B and C)

    InvRotatA = ginv(optrotationA$RotationMatrix)
    rotatedCore = InvRotatA %*% Sol$H %*%
    kronecker(t(ginv(optrotationC$RotationMatrix)),
    t(ginv(optrotationB1$RotationMatrix)))

    # counterrotation for E (based on optimal rotation for A)

    rotationB2 = t(InvRotatA %*%
    t(Sol$B2))
```

```
        optrotationB2 = procr(Sol$B2, TrueSol$B2[, 1:r1])

        #to find optimal E rotation

        rotationA = Sol$A %*% ginv(optrotationB2$RotationMatrix)

        #although this is not very relevant

    }

    # compute Tucker congruence coefficients

    if (SeparateAnalysis == 1) {
        Out$TuckerA1 = optrotationA1$TuckerCongruence
    } else {
        Out$TuckerA = optrotationA$TuckerCongruence
    }
    Out$TuckerB1 = optrotationB1$TuckerCongruence
    Out$TuckerC = optrotationC$TuckerCongruence
    Out$TuckerCore = sum((rotatedCore - TrueSol$CoreMatrix[1:r1, ])^2)
    Out$TuckerCoreSize = Out$TuckerCore/(r1 * r2 * r3)

    if (SeparateAnalysis == 1) {
        Out$TuckerA2optimal = optrotationA2$TuckerCongruence
        Out$TuckerB2 = ComputeTuckerMatrix(rotationB2, TrueSol$B2[, 1:r1])
        Out$TuckerB2optimal = optrotationB2$TuckerCongruence
        Out$TuckerA2 = ComputeTuckerMatrix(rotationA2, TrueSol$A2[, 1:r1])
    } else {
        Out$TuckerB2 = ComputeTuckerMatrix(rotationB2, TrueSol$B2[, 1:r1])
        Out$TuckerB2optimal = optrotationB2$TuckerCongruence
    }
}
```

```
    return(Out)
}
```

7.4.2. ComputeRecoveryT3PCAExtra.R

ComputeRecoveryT3PCAExtra.R calcula las congruencias entre la solución real (cuando se ha añadido una componente extra adicional) y la solución T3-PCA obtenida por el algoritmo.

```
ComputeRecoveryT3PCAExtra <- function(TrueSol, Sol, SeparateAnalysis,
ProxyFit) {
    # compute congruences between true (with extra component)
    # and obtained T3PCA-solution

    # SeparateAnalysis: Sol obtained from the separate analysis strategy?
    # 0 = no (Sol from simultaneous or segmented strategy)
    # 1 = yes (Sol from separate analysis)
    # Extracomp: solution with extra component
    # 0 = no (no extra component) 1 = yes

    library(MASS)
    source("procr.R")
    source("ComputeTuckerMatrix.R")

    checkinput = 1

    if ((SeparateAnalysis != 0) & (SeparateAnalysis != 1)) {
        cat(" ", fill = TRUE)
        cat("SeparateAnalysis should be 0 (simultaneous/segmented)
        or 1 (separate)",fill = TRUE)
        cat(" ", fill = TRUE)
        checkinput = 0
```

```
}

Out = list()

if (checkinput == 1) {

    # Comparing badness-of-fit and badness-of-data

    Out$BOFBOD = Sol$Fit - TrueSol$BODExtra
    Out$BOFBOD3D = Sol$Fit3D - TrueSol$BOD3DExtra
    Out$BOFBOD2D = Sol$Fit2D - TrueSol$BOD2DExtra
    Out$BOFBODsize = Sol$Fitsize - TrueSol$BODsizeExtra
    Out$BOFBOD3Dsize = Sol$Fit3Dsize - TrueSol$BOD3DsizeExtra
    Out$BOFBOD2Dsize = Sol$Fit2Dsize - TrueSol$BOD2DsizeExtra

    Out$BODreached = Sol$Fit <= ProxyFit
    Out$BODdiff = Sol$Fit - ProxyFit
    Out$nRunsOptimalFit = sum(Sol$FitValues == min(Sol$FitValues))

    # Recovery of M (BOR)

    r1 = Sol$Info$RankVector[1]
    r2 = Sol$Info$RankVector[2]
    r3 = Sol$Info$RankVector[3]

    if (SeparateAnalysis == 1) {
        Model3D = Sol$A1 %*% matrix(Sol$H, r1, r2 * r3) %*%
        kronecker(t(Sol$C),t(Sol$B1))
        Model2D = Sol$A2 %*% t(Sol$B2)
    } else {
        Model3D = Sol$A %*% matrix(Sol$H, r1, r2 * r3) %*%
        kronecker(t(Sol$C),t(Sol$B1))
```

```
    Model2D = Sol$A %*% t(Sol$B2)
}

Out$Recov = 0
Out$Recov3D = sum((Model3D - TrueSol$TrueWay3DMatrix)^2)
Out$Recov2D = sum((Model2D - TrueSol$TrueWay2D)^2)
Out$Recov = (Sol$Info$Alfa * Out$Recov3D) + ((1 - Sol$Info$Alfa) * Out$Recov2
Out$Recovsize = Out$Recov/((Sol$Info$nRows * Sol$Info$nColumns3D *
Sol$Info$nSlices) +(Sol$Info$nRows * Sol$Info$nColumns2D))
Out$Recov3Dsize = Out$Recov3D/(Sol$Info$nRows * Sol$Info$nColumns3D *
Sol$Info$nSlices)
Out$Recov2Dsize = Out$Recov2D/(Sol$Info$nRows * Sol$Info$nColumns2D)

if (SeparateAnalysis == 1) {

    # Recovery measures for component matrices

    optrotationA1 = procr(Sol$A1, TrueSol$A1)
    optrotationB1 = procr(Sol$B1, TrueSol$B1)
    optrotationC = procr(Sol$C, TrueSol$C)

    # counterrotation for the core
    # (based on optimal rotations for A, B and C)

    InvRotatA1 = ginv(optrotationA1$RotationMatrix)
    rotatedCore = InvRotatA1 %*% Sol$H %*%
    kronecker(t(ginv(optrotationC$RotationMatrix)),
    t(ginv(optrotationB1$RotationMatrix)))

    # counterrotation for E (based on optimal rotation for A)

    optrotationA2 = procr(Sol$A2, TrueSol$A2)
```

```
    InvRotatA2 = ginv(optrotationA2$RotationMatrix)
    rotationB2 = t(InvRotatA2 %*% t(Sol$B2))
    optrotationB2 = procr(Sol$B2, TrueSol$B2)
    InvRotatB2 = ginv(optrotationB2$RotationMatrix)
    rotationA2 = Sol$A2 %*% InvRotatB2
} else {

    # Recovery measures for component matrices

    optrotationA_3D = procr(Sol$A, TrueSol$A1)
    optrotationA_2D = procr(Sol$A, TrueSol$A2)
    optrotationB1 = procr(Sol$B1, TrueSol$B1)
    optrotationC = procr(Sol$C, TrueSol$C)

    # counterrotation for the core
    # (based on optimal rotations for A, B and C)

    InvRotatA_3D = ginv(optrotationA_3D$RotationMatrix)
    InvRotatA_2D = ginv(optrotationA_2D$RotationMatrix)
    rotatedCore_3D = InvRotatA_3D %*% Sol$H %*%
    kronecker(t(ginv(optrotationC$RotationMatrix)),
    t(ginv(optrotationB1$RotationMatrix)))
    rotatedCore_2D = InvRotatA_2D %*% Sol$H %*%
    kronecker(t(ginv(optrotationC$RotationMatrix)),
    t(ginv(optrotationB1$RotationMatrix)))

    # counterrotation for E (based on optimal rotation for A)

    rotationB2_3D = t(InvRotatA_3D %*% t(Sol$B2))
    rotationB2_2D = t(InvRotatA_2D %*% t(Sol$B2))
    optrotationB2 = procr(Sol$B2, TrueSol$B2)
```

```
        #to find optimal E rotation

        rotationA = Sol$A %*% ginv(optrotationB2$RotationMatrix)
        #although this is not very relevant
    }

    # compute Tucker congruence coefficients

    if (SeparateAnalysis == 1) {
        Out$TuckerA1 = optrotationA1$TuckerCongruence
        Out$TuckerB1 = optrotationB1$TuckerCongruence
        Out$TuckerC = optrotationC$TuckerCongruence
        Out$TuckerCore = sum((rotatedCore - TrueSol$CoreMatrix)^2)
        Out$TuckerCoreSize = Out$TuckerCore/(r1 * r2 * r3)
        Out$TuckerA2optimal = optrotationA2$TuckerCongruence
        Out$TuckerB2 = ComputeTuckerMatrix(rotationB2, TrueSol$B2)
        Out$TuckerB2optimal = optrotationB2$TuckerCongruence
        Out$TuckerA2 = ComputeTuckerMatrix(rotationA2, TrueSol$A2)
    } else {
        Out$TuckerA_3D = optrotationA_3D$TuckerCongruence
        Out$TuckerA_2D = optrotationA_2D$TuckerCongruence
        Out$TuckerB1 = optrotationB1$TuckerCongruence
        Out$TuckerC = optrotationC$TuckerCongruence
        Out$TuckerCore_3D = sum((rotatedCore_3D - TrueSol$CoreMatrix)^2)
        Out$TuckerCoreSize_3D = Out$TuckerCore_3D/(r1 * r2 * r3)
        Out$TuckerCore_2D = sum((rotatedCore_2D - TrueSol$CoreMatrix)^2)
        Out$TuckerCoreSize_2D = Out$TuckerCore_2D/(r1 * r2 * r3)
        Out$TuckerB2_3D = ComputeTuckerMatrix(rotationB2_3D, TrueSol$B2)
        Out$TuckerB2_2D = ComputeTuckerMatrix(rotationB2_2D, TrueSol$B2)
        Out$TuckerB2optimal = optrotationB2$TuckerCongruence
    }
}
```

```
    return(Out)
}
```

7.5. SIMULACIÓN: doTucker3PCAsimulation.R

Para llevar a cabo la simulación se utiliza la función **'doTucker3PCAsimulation.R'**.

En el estudio de simulación, en primer lugar se generan los datos (**'GenerateTucker3-PCA.R'**), se preprocesan (**'PreProcess3D.R'**) y a continuación los datos generados se analizan mediante (1) el algoritmo integrado T3-PCA (**'T3PCAfunc.R'**), (2) una estrategia segmentada que aplica primero T3 y después PCA (**'T3PCASegmented.R'**), (3) una estrategia segmentada que aplica primero PCA y después T3 (**'T3PCASegmented.R'**) , (4) una estrategia separada: T3 sobre el bloque 3D e independientemente PCA sobre el bloque 2D (**'T3PCAseparate.R'**). Finalmente se calcula el recovery para cada uno de los cuatro análisis anteriores (**'ComputeRecoveryT3-PCA.R'**).

En la Figura 7.6 se muestra un esquema del algoritmo.

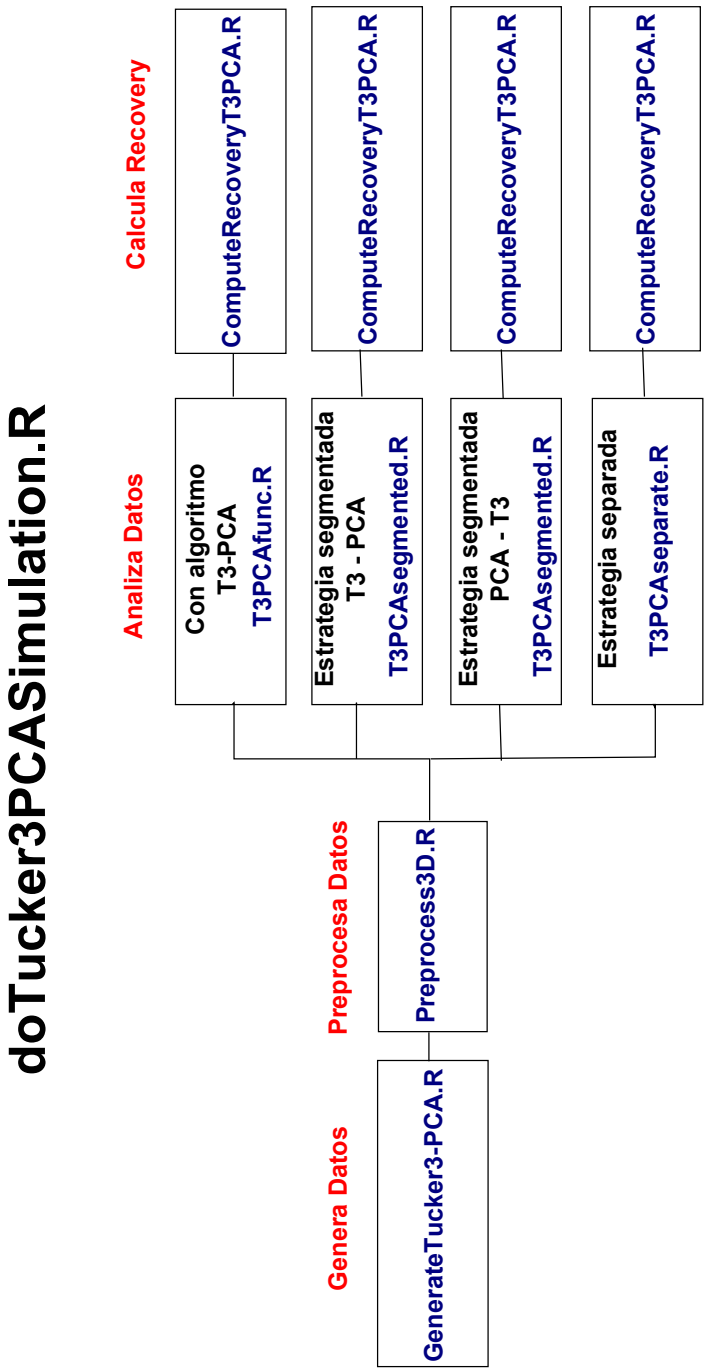


Figura 7.6: Esquema del Algoritmo de Simulacion

El código del programa **'doTucker3PCAsimulation.R'** es el siguiente:

```
doTucker3PCAsimulationCluster <- function(Args) {
    OriginalAlfa = cbind(0.999, 0.95, 0.9, 0.85, 0.8, 0.75, 0.7, 0.6, 0.5 ,
    0.4, 0.3, 0.2, 0.1, 0.05, 0.001)

    s_nRow = Args[1]
    s_nCol3D = Args[2]
    s_nSlice = Args[3]
    s_nCol2D = Args[4]
    s_r1 = Args[5]
    s_r2 = Args[6]
    s_r3 = Args[7]
    s_nExtraComponents = Args[8]
    s_ExplVarExtraComp = Args[9]   # 0-1000
    s_noise = Args[10]   # 0-1000
    s_preprocess = Args[11]   # 0 or 1
    s_AlternativeLossF = Args[12]
    s_conv = Args[13]   # 0-100000000
    s_nRuns = Args[14]
    s_GenerSeed = Args[15]
    s_AnalSeed = Args[16]
    s_SimulNr = Args[17]

    nRow = as.numeric(s_nRow)
    nCol3D = as.numeric(s_nCol3D)
    nSlice = as.numeric(s_nSlice)
    nCol2D = as.numeric(s_nCol2D)
    r1 = as.numeric(s_r1)
    r2 = as.numeric(s_r2)
    r3 = as.numeric(s_r3)
    nExtraComponents = as.numeric(s_nExtraComponents)
    tempExplVarExtraComp = as.numeric(s_ExplVarExtraComp)
```

```
ExplVarExtraComp = tempExplVarExtraComp/1000
tempnoise = as.numeric(s_noise)
noise = tempnoise/1000
preprocess = as.numeric(s_preprocess)
AlternativeLossF = as.numeric(s_AlternativeLossF)
tempconv = as.numeric(s_conv)
conv = tempconv/1e+08
nRuns = as.numeric(s_nRuns)
GenerSeed = as.numeric(s_GenerSeed)
AnalSeed = as.numeric(s_AnalSeed)
SimulNr = as.numeric(s_SimulNr)

if (r1 == 1) {
    tt = cbind(1) * (1 - ExplVarExtraComp)
} else {
    if (r1 == 2) {
        tt = cbind(0.7, 0.3) * (1 - ExplVarExtraComp)
    } else {
        if (r1 == 3) {
            tt = cbind(0.6, 0.25, 0.15) * (1 - ExplVarExtraComp)
        } else {
            if (r1 == 4) {
              tt = cbind(0.5, 0.3, 0.1, 0.1) * (1 - ExplVarExtraComp)
            }
        }
    }
}

ExplVarVector = cbind(tt, ExplVarExtraComp)
rm(tt)

source("GenerateSeed.R")
```

```
source("PreProcess3D.R")
source("GenerateTucker3_PCA.R")
source("T3PCAfunc.R")
source("T3PCAsegmented.R")
source("T3PCAseparate.R")
source("ComputeRecoveryT3PCA.R")
source("ComputeRecoveryT3PCAExtra.R")
```

Realiza una llamada a la función $GenerateTucker3-PCA$ para generar las matrices de 3 y 2 vías: Way3D, Way2D.

```
cputime = system.time({
    TrueSol = GenerateTucker3_PCA(nRow, nCol3D, nSlice, nCol2D, r1, r2,
        r3, noise, ExplVarVector, OriginalAlfa[1], AlternativeLossF,
        nExtraComponents,
        GenerSeed)
```

Se preprocesan las matrices Way3D y Way2D

```
    if (preprocess == 1) {
        Way3D = PreProcess3D(TrueSol$Way3D)
        Way2D = scale(TrueSol$Way2D, T, T)

        #standardization of the variables (center and normalize)

    } else {
        Way3D = TrueSol$Way3D
        Way2D = TrueSol$Way2D
    }
    TrueSol$preprocess = preprocess
```

```
if (AlternativeLossF == 1) {
    ssq3D = sum(TrueSol$Way3D^2)
    ssq2D = sum(TrueSol$Way2D^2)
}

AnalysisSeeds = GenerateSeed(8 * length(OriginalAlfa), AnalSeed)
```

Se realiza el estudio de simulación para cada uno de los valores de alfa considerados

```
for (alfatel in 1:length(OriginalAlfa)) {
    if (AlternativeLossF == 1) {
        Alfa = (OriginalAlfa[alfatel] * ssq2D) /((OriginalAlfa[alfatel] *
          ssq2D) + ((1 - OriginalAlfa[alfatel]) * ssq3D))
    } else {
        Alfa = OriginalAlfa[alfatel]
    }

    TrueSol$BOD[alfatel] = (Alfa * TrueSol$BOD3D) + ((1 - Alfa)
    * TrueSol$BOD2D)
    TrueSol$BODsize[alfatel] = TrueSol$BOD[alfatel]/((nRow * nCol3D *
        nSlice) + (nRow * nCol2D))
    TrueSol$BODExtra[alfatel] = (Alfa * TrueSol$BOD3DExtra) + ((1 -
        Alfa) * TrueSol$BOD2DExtra)
    TrueSol$BODsizeExtra[alfatel] = TrueSol$BODExtra[alfatel]/((nRow *
        nCol3D * nSlice) + (nRow * nCol2D))
    rm(Alfa)
```

Se realiza el estudio de simulación sin añadir una componente adicional

```
# analysis without extra component(s)

SolSimul[alfatel] = T3PCAfunc(Way3D, Way2D, nRow, nCol3D, nSlice,
    nCol2D, r1, r2, r3, conv, OriginalAlfa[alfatel],
    AlternativeLossF, nRuns, AnalysisSeeds[((alfatel -1) * 8) + 1])
SolSegmented3D[alfatel] = T3PCAsegmented(Way3D, Way2D, r1, r2, r3,
    conv, OriginalAlfa[alfatel], AlternativeLossF, nRuns, 1
    , AnalysisSeeds[((alfatel -1) * 8) + 2])
SolSegmented2D[alfatel] = T3PCAsegmented(Way3D, Way2D, r1, r2, r3,
    conv, OriginalAlfa[alfatel], AlternativeLossF, nRuns, 2
    , AnalysisSeeds[((alfatel -1) * 8) + 3])
SolSeparate[alfatel] = T3PCAseparate(Way3D, Way2D, r1, r2, r3,
    conv, OriginalAlfa[alfatel], AlternativeLossF, nRuns,
    AnalysisSeeds[((alfatel -1) * 8) + 4])

ProxyFit[alfatel] = min(SolSimul[alfatel]$Fit,
 SolSegmented3D[alfatel]$Fit,SolSegmented2D[alfatel]$Fit,
  TrueSol$BOD[alfatel])
RecovSimul[alfatel] = ComputeRecoveryT3PCA(TrueSol,
SolSimul[alfatel],0, ProxyFit[alfatel])
RecovSegmented3D[alfatel] = ComputeRecoveryT3PCA(TrueSol,
 SolSegmented3D[alfatel],0, ProxyFit[alfatel])
 RecovSegmented2D[alfatel] = ComputeRecoveryT3PCA(TrueSol,
 SolSegmented2D[alfatel],0, ProxyFit[alfatel])
RecovSeparate[alfatel] = ComputeRecoveryT3PCA(TrueSol, 7
SolSeparate[alfatel],1, ProxyFit[alfatel])

labs = cbind("", "A1", "B1", "C", "Core", "A2", "B2")
rec1 = cbind("simultan", RecovSimul[alfatel]$TuckerA,
RecovSimul[alfatel]$TuckerB1,
RecovSimul[alfatel]$TuckerC,
RecovSimul[alfatel]$TuckerCoreSize,
```

```
RecovSimul[alfatel]$TuckerA,
RecovSimul[alfatel]$TuckerB2optimal)
rec2 = cbind("segmen3D",RecovSegmented3D[alfatel]$TuckerA,
RecovSegmented3D[alfatel]$TuckerB1,
RecovSegmented3D[alfatel]$TuckerC,
RecovSegmented3D[alfatel]$TuckerCoreSize,
RecovSegmented3D[alfatel]$TuckerA,
RecovSegmented3D[alfatel]$TuckerB2optimal)
rec3 = cbind("segmen2D", RecovSegmented2D[alfatel]$TuckerA,
RecovSegmented2D[alfatel]$TuckerB1,
RecovSegmented2D[alfatel]$TuckerC,
RecovSegmented2D[alfatel]$TuckerCoreSize,
RecovSegmented2D[alfatel]$TuckerA,

RecovSegmented2D[alfatel]$TuckerB2optimal)
rec4 = cbind("separate", RecovSeparate[alfatel]$TuckerA1,
RecovSeparate[alfatel]$TuckerB1,
RecovSeparate[alfatel]$TuckerC,
RecovSeparate[alfatel]$TuckerCoreSize,
RecovSeparate[alfatel]$TuckerA2optimal,

RecovSeparate[alfatel]$TuckerB2optimal)
RecovSummary[alfatel] = t(cbind(t(labs), t(rec1), t(rec2),
 t(rec3),t(rec4)))
rm(labs, rec1, rec2, rec3, rec4)
```

Se realiza el estudio de simulación añadiendo una componente adicional

```
# analysis with extra component(s)
```

```
SolSimulExtra[alfatel] = T3PCAfunc(Way3D, Way2D, nRow, nCol3D,
    nSlice,nCol2D, r1 + nExtraComponents, r2, r3, conv,
    OriginalAlfa[alfatel],AlternativeLossF, nRuns,
    AnalysisSeeds[((alfatel - 1) * 8) + 5])
SolSegmented3DExtra[alfatel] = T3PCAsegmented(Way3D, Way2D, r1 +
    nExtraComponents, r2, r3, conv, OriginalAlfa[alfatel],
    AlternativeLossF,nRuns, 1, AnalysisSeeds[((alfatel -1) * 8)
    + 6])
SolSegmented2DExtra[alfatel] = T3PCAsegmented(Way3D, Way2D, r1 +
    nExtraComponents, r2, r3, conv, OriginalAlfa[alfatel],
    AlternativeLossF,nRuns, 2, AnalysisSeeds[((alfatel -1) * 8)
    + 7])
SolSeparateExtra[alfatel] = T3PCAseparate(Way3D, Way2D, r1 +
 nExtraComponents,r2, r3, conv, OriginalAlfa[alfatel],
 AlternativeLossF, nRuns,
    AnalysisSeeds[((alfatel - 1) * 8) + 8])

ProxyFitExtra[alfatel] = min(SolSimulExtra[alfatel]$Fit,
 SolSegmented3DExtra[alfatel]$Fit,
    SolSegmented2DExtra[alfatel]$Fit, TrueSol$BODExtra[alfatel])
RecovSimulExtra[alfatel] = ComputeRecoveryT3PCAExtra(TrueSol,
 SolSimulExtra[alfatel],0, ProxyFitExtra[alfatel])
RecovSegmented3DExtra[alfatel] = ComputeRecoveryT3PCAExtra(TrueSol,
    SolSegmented3DExtra[alfatel], 0, ProxyFitExtra[alfatel])
RecovSegmented2DExtra[alfatel] = ComputeRecoveryT3PCAExtra(TrueSol,
    SolSegmented2DExtra[alfatel], 0, ProxyFitExtra[alfatel])
RecovSeparateExtra[alfatel] = ComputeRecoveryT3PCAExtra(TrueSol,
    SolSeparateExtra[alfatel], 1, ProxyFitExtra[alfatel])

labsExtra = cbind("", "A1", "B1", "C", "Core", "A2", "B2")
rec1Extra = cbind("simultan",
RecovSimulExtra[alfatel]$TuckerA_3D,
```

```
            RecovSimulExtra[alfatel]$TuckerB1,
            RecovSimulExtra[alfatel]$TuckerC,
            RecovSimulExtra[alfatel]$TuckerCoreSize_3D,
            RecovSimulExtra[alfatel]$TuckerB2optimal)
            rec2Extra = cbind("segmen3D",
            RecovSegmented3DExtra[alfatel]$TuckerA_3D,
            RecovSegmented3DExtra[alfatel]$TuckerB1,
            RecovSegmented3DExtra[alfatel]$TuckerC,
            RecovSegmented3DExtra[alfatel]$TuckerCoreSize_3D,
            RecovSegmented3DExtra[alfatel]$TuckerA_3D,
            RecovSegmented3DExtra[alfatel]$TuckerB2optimal)
            rec3Extra = cbind("segmen2D",
            RecovSegmented2DExtra[alfatel]$TuckerA_3D,
            RecovSegmented2DExtra[alfatel]$TuckerB1,
            RecovSegmented2DExtra[alfatel]$TuckerC,
            RecovSegmented2DExtra[alfatel]$TuckerCoreSize_3D,
            RecovSegmented2DExtra[alfatel]$TuckerA_3D,
            RecovSegmented2DExtra[alfatel]$TuckerB2optimal)
            rec4Extra = cbind("separate",
            RecovSeparateExtra[alfatel]$TuckerA1,
            RecovSeparateExtra[alfatel]$TuckerB1,
            RecovSeparateExtra[alfatel]$TuckerC,
            RecovSeparateExtra[alfatel]$TuckerCoreSize,
            RecovSeparateExtra[alfatel]$TuckerA2optimal,
            RecovSeparateExtra[alfatel]$TuckerB2optimal)
            RecovSummaryExtra[alfatel] = t(cbind(t(labsExtra), t(rec1Extra),
                t(rec2Extra), t(rec3Extra), t(rec4Extra)))
            rm(labsExtra, rec1Extra, rec2Extra, rec3Extra, rec4Extra)
        }
    })

    Out = list()
```

```
    Out$TrueSol = TrueSol
    Out$SolSimul = SolSimul
    Out$SolSegmented3D = SolSegmented3D
    Out$SolSegmented2D = SolSegmented2D
    Out$SolSeparate = SolSeparate
    Out$RecovSimul = RecovSimul
    Out$RecovSegmented3D = RecovSegmented3D
    Out$RecovSegmented2D = RecovSegmented2D
    Out$RecovSeparate = RecovSeparate
    Out$RecovSummary = RecovSummary
    Out$SolSimulExtra = SolSimulExtra
    Out$SolSegmented3DExtra = SolSegmented3DExtra
    Out$SolSegmented2DExtra = SolSegmented2DExtra
    Out$SolSeparateExtra = SolSeparateExtra
    Out$RecovSimulExtra = RecovSimulExtra
    Out$RecovSegmented3DExtra = RecovSegmented3DExtra
    Out$RecovSegmented2DExtra = RecovSegmented2DExtra
    Out$RecovSeparateExtra = RecovSeparateExtra
    Out$RecovSummaryExtra = RecovSummaryExtra
    Out$Time = list()
    Out$Time$TotalTime = round(cputime[1], 2)
    return(Out)
}
```

CONCLUSIONES

Una exhaustiva revisión de la bibliografía especializada en el campo de la Interacción en Tablas de Dos Vías, los Modelos para Tres Vías y Modelos para Datos Acoplados, nos ha permitido:

(1) Conocer el impacto que tienen los métodos multivariantes descriptivos en el Análisis de la interacción en tablas de dos vías, en concreto los modelos AMMI y GGE son los más utilizados en la evaluación de cultivos. Al mismo tiempo se propone un nuevo paquete en el lenguaje `R` llamado `GGEbiplotGUI`, que implementa las principales funcionalidades de los métodos biplot con la ventaja de ser software libre y además disponer de una interfaz gráfica que no requiere de conocimientos de `R` para su uso.

(2) Conocer los Modelos para el Análisis de Tablas de Tres Vías de tipo continuo, realizando un exhaustivo estudio de los mismos, especialmente de los Modelos $Tucker3$ y $CANDECOMP/PARAFAC$, lo cual nos ha permitido estudiar las diferencias entre los mismos y conocer situaciones en las que son útiles cada uno de ellos.

(3) Conocer los Modelos para el Análisis de Tablas de Tres Vías de tipo binario, realizando un exhaustivo estudio de los mismo, especialmente de los Modelos $INDCLAS$ y $Tucker3$-$HICLAS$.

(4) Introducirnos en el estudio de Datos Acoplados presentando dos de los modelos existentes para el análisis de los mismos: $PARAFAC$-PCA y $CHIC$.

En esta tesis se propone T3-PCA, un modelo global para el análisis integrado de un conjunto de datos acoplados, consistente en un array de datos continuos en tres vías y una matriz de datos continuos que comparten un modo. En el modelo T3-PCA se utiliza el modelo Tucker3 para modelizar el array de tres vías y el modelo PCA para modelar la matriz de dos vías. Para este nuevo modelo podemos afirmar:

(5) Se ha propuesto un algoritmo para el cálculo de los parámetros del modelo y un método para la selección del número de componentes a retener.

(6) Dado que el modelo $Tucker3$ y el modelo PCA tienen libertad rotacional, el modelo combinado $T3$-PCA también la tiene. De manera que podemos rotar las matrices de componentes y la core-matrix para obtener una solución más facilmente interpretable.

(7) El modelo $T3$-PCA utiliza la información de los dos bloques de datos para estimar el modo común. Este modelo es menos restrictivo que el modelo $PARAFAC$-PCA ya que el número de componentes no está restringido a ser igual en los tres modos y se permiten interacciones entre todas las componentes de los distintos modos.

(8) El algoritmo $T3$-PCA es capaz de identificar el mínimo global de la función de pérdida. Sólo en 27 de los 9.600 conjuntos de datos el valor de la función de pérdida obtenida aplicando el algoritmo es mayor que el valor proxy. Además en todos estos casos el valor de alfa es 0,999.

(9) El algoritmo $T3$-PCA, según ha quedado demostrado en el estudio de simulación, recupera de forma satisfactoria los parámetros del modelo. Además el algoritmo $T3$-PCA supera a la estrategia segmentada (en la que los componentes comunes se determinan usando solo la información contenida en uno de los bloques de datos).

(10) Al usar la información contenida en ambos bloques de datos, la estrategia integrada recupera mejor las componentes comunes e ignora las componentes que son específicas de uno de los bloques de datos (i.e., las componentes distintivas). Los mejores resultados del algorítmo $T3$-PCA se obtienen cuando ambos bloques de datos tienen igual influencia en el análisis (i.e., $\alpha = 0,5$).

Líneas de investigación futuras

Obviamente el modelo $T3$-PCA puede ser extendido y restringido de diferentes formas:

(1) Una posible mejora podría ser realizar una rotación combinada. Kiers (1998) desarrolló un procedimiento en el cual se puede dar un peso a cada matriz de componentes y a la core, este peso indica la importancia que la matriz/core tiene a la hora de la interpretación. Para el caso del modelo $T3$-PCA sería interesante desarrollar un procedimiento similar en el que el usuario pueda decidir el peso de $\mathbf{A}$, $\mathbf{B}^1$, $\mathbf{C}$, $\mathbf{B}^2$ y $\underline{\mathbf{G}}$.

(2) Algunas posibles extensiones del modelo serían:

   a) Tener información de dos vías para algún modo adicional

   b) Tener múltiples bloques de dos vías para un modo

   c) Imponer restricciones, por ejemplo no negatividad (cuando la matriz de tres vías son niveles de emoción a lo largo del tiempo para diferentes personas, las componentes no deberían ser negativas)

ARTÍCULOS

Artículos:

a) An interactive biplot implementation in R for modeling genotype-by-environment interaction. Frutos E and Galindo MP and Leiva V. *Stochastic Environmental Research and Risk Assessment* **2013**.

   Esta revista está indexada en **Journal Citation Report**. Tiene un factor de impacto de 2,673 y pertenece a las siguientes categorias:

   a) ENGINEERING, CIVIL. Ranking: 7/124 (1er cuartil)

   b) ENGINEERING, ENVIRONMENTAL. Ranking: 17/44 (2do cuartil)

   c) SCIENCES. Ranking: 64/215 (2do cuartil)

   d) STATISTICS and PROBABILITY. Ranking: 6/119 (1er cuartil)

   e) WATER RESOURCES. Ranking: 11/79 (1er cuartil)

b) Data fusion by $T3$-PCA: An integrated global model for the simultaneous analysis of coupled real-valued data. Sometido a la revista *Behavior Research Methods*, indexada en **Journal Citation Report** y con índice de impacto 2,458.

Stoch Environ Res Risk Assess
DOI 10.1007/s00477-013-0821-z

REVIEW PAPER

An interactive biplot implementation in R for modeling genotype-by-environment interaction

Elisa Frutos · M. Purificación Galindo · Víctor Leiva

Abstract Classical and GGE biplot methods are graphical procedures that allow multivariate data to be analyzed. In particular, the GGE biplot displays the genotype main effect (G) and the genotype by environment interaction (GE) in two-way data. The GGE biplot originates from data graphical analysis of multi-environment trials (MET). Thus, agronomists, crop scientists and geneticists are potential users of this method. However, it can also be used to visualize and analyze other types of data. In this paper, we propose a new interactive computational implementation in `R` language to perform the main functions of the classical and GGE biplot methods, so it is also useful for MET data visual analysis. This implementation is organized in an `R` package named `GGEBiplotGUI`. This package is the only interactive, noncommercial and open source software that currently exists, offering a free alternative to available commercial software. In addition, it can be used without to practically have knowledge of the `R` programming language. Here, we present and discuss the capabilities and features of the `GGEBiplotGUI` package and illustrate them by using real data. The `GGEBiplotGUI` package graphically addresses the questions that a researcher likely asks. This `R` package is not only a tool for visual data analysis of multi-environment trials, useful for plant breeders and geneticists, in order to study yields from genotypes and interactions between genotype and environment, but also data from other areas can be analyzed by the `GGEBiplotGUI` package.

Keywords Graphical display · Least squares method · Multivariate data analysis · Singular value decomposition · Statistical models · Statistical software

Mathematics Subject Classification 62H99 · 62-04

1 Introduction and bibliographical review

The biplot method is a graphical display of multivariate data. This method was introduced by Gabriel (1971) in the context of principal component analysis (PCA). Specifically, the biplot is a joint graphical representation, in a low dimensional Euclidean space (usually a plane), of a multivariate data matrix by markers for its rows and columns, chosen in such a way that the inner (or scalar) product represents the elements of the data matrix. Due to the properties of the inner product, the biplot is a powerful data visualization tool, which can be viewed as a multivariate extension of the scatterplot, because the biplot representation is usually performed in the two-dimensional space of the set of real numbers. This is the classical focus of the biplot method (classical biplot), which has two parts. First, it carries out the approximation of the data matrix by a singular value decomposition (SVD) and, second, this matrix is factorized in row and column markers; see Eckart and Young (1939) and Golub and Reinsch (1970).

Gower and Harding (1988), Gower (1992), and Gower and Hand (1996) provided a different focus of the biplot to

E. Frutos · M. P. Galindo
Departamento de Estadística, Universidad de Salamanca, Salamanca, Spain

V. Leiva (✉)
Departamento de Estadística, Universidad de Valparaíso, Avda. Gran Bretaña 1111, Playa Ancha, Valparaiso, Chile
e-mail: victor.leiva@uv.cl
URL: http://www.deuv.cl/leiva

Published online: 16 November 2013

that originally proposed by Gabriel (1971). They first ordered the individuals using scaling, and then superposed the variables in a such a way that a joint graphical interpretation is possible, as usual in biplot methods. Both of these representations are just descriptive, so that no parametrical assumptions are considered. For more details about the biplot methods, see Gower and Hand (1996), which is recognized as the first book on this method, and Greenacre (2010) and Gower et al. (2011), for more recent books.

Another focus different from the classical one is based on the relation between the approximation of least squares (LS) of matrices and the SVD. This third focus permits the biplots to be visualized as models fitted by bilinearity. These bilinear models are known as regression biplots and can be interpreted as a multiplicative bilinear model, allowing us to demonstrate in graphical form the association, linear or not, between subjects and/or variables; see Gabriel (1998) and Vicente-Villardón et al. (2006). In this line, a number of authors have studied the case when the biplots are used to describe interaction between row and column effects in multiplicative and additive bilinear models; see, e.g., Denis (1991), Falguerolles (1995), Van Eeuwijk (1995), and Choulakian (1966). In order to approximate the biplots in the case of these last models, the mentioned authors used estimation methods employed in generalized bilinear models, which are considered as an extension of the generalized linear models; see Nelder and Wedderburn (1972). An exhaustive study on all of these alternative ways of the biplot method can be seen in Cárdenas and Galindo (2003).

Since Gabriel (1971)'s paper, the biplot method has increasingly been used in data visualization and analysis in diverse disciplines, being it particularly useful in exploring data from areas such as agricultural, ecological and environmental sciences, and genetics. In these areas, the size of the data sets often result to be large, with complicated interactions and interconnections. These sets are typically genotype-by-environment-by-trait three-way data, which provide rich information. Such data can be organized into several types of two-way tables, as for example: (i) genotype × environment tables for each trait; (ii) genotype × trait tables for each environment, across subsets of environments, and across all environments; (iii) environment × trait tables for each genotype, across subsets of genotypes, and across all genotypes; (iv) phenotype (genotype–environment combination) × trait tables; (v) genotype × environment-trait tables; and (vi) combined two-way tables of genotype × explanatory variables plus response variables. A full understanding of the type of data requires knowledge of all these two-way tables. From the viewpoint of genotype evaluation, genotype by environment and genotype by trait tables are the most relevant; see Yan et al. (2000).

We focus our attention on data of genotype by environment, which provide rich information for different purposes, but we want to stress the methodology discussed in the paper can also be applied to several other types of data, for example in geosciences; see Bacon-Shone (2009). Plant geneticists often study the performance of many genotypes in diverse environments. Such studies are conducted in order to select the best genotypes for improvements of crops. The data collected for these studies correspond to one or more attributes, for each genotype in each environment. This type of data can be analyzed in a straightforward way by means of analysis of variance (ANOVA) models. However, other models can also be used. Particularly, the additive main effects and multiplicative interaction (AMMI) model and genotype main effects (G) and genotype × environment interaction effects (GE) model, known as GGE, are the most used to describe the kind of data above mentioned; see Gauch (1988). The AMMI model (AMMI biplot) has been extensively applied in the statistical analysis of multiple-environment trials (MET); see Kempton (1984), Gauch and Zobel (1996, 1997), and Ebdon and Gauch (2002a, b). The GGE model (GGE biplot) was proposed by Yan et al. (2000), allowing genotype by environment interaction (GEI) of MET data to be visually examined. Several recent papers have exhaustively compared and contrasted AMMI and GGE models, with respect to their suitability for genotype by environment interaction analysis; see Gauch (2006), Yan and Tinker (2006), Yan et al. (2007) and Gauch et al. (2008). In AMMI models, as multiplicative terms as needed are incorporated in order to explain the variability of the second order interaction. Such models are based on the decomposition in singular values and vectors of the residual matrix of the associated linear model. The GGE model applies the SVD to the data, subtracting the environment effects, because these biplots display both G and GE, which are the two sources of variation relevant to cultivar evaluation; see Kang (1988), Gauch and Zobel (1996) and Yan and Kang (2003). This model is based on multiplicative linear-bilinear site regression; see Cornelius et al. (1996).

Bibliography on biplots is generous, with many papers published in various scientific journals, whereas about 50,000 websites containing the word biplot are available in the internet. Macros for biplot analysis have been implemented in the main commercial statistical softwares; see, e.g., Yan and Tinker (2006). Currently, most commercial statistical software packages include a procedure or macro for generating biplots. Specifically, the `GGEbiplot` program, dedicated to the GGE biplot (see http://www.ggebiplot.com), can also generate the classical biplot. This program is a commercial software and is widely used by agronomists, crop scientists and geneticists; see Yan and Kang (2003, 2006). However, today the scientific community has at its disposal a non-commercial and open

source software for statistics and graphs, named R, which can be obtained at no cost from http://www.r-project.org. The statistical software R is currently very popular in the international scientific community; see R Team (2013), and Leiva et al. (2008) and Barros et al. (2009), for some R packages that have been developed. Thus, it seems to be necessary to count with an R package that offers a non-commercial and open source software as alternative to the existing commercial software on biplots. This R package should address the main questions that a researcher could ask and have an interactive characteristic, because in this way it can be more attractive for practitioners that does not know the R programming language.

The aims of this work are (i) to propose a new interactive computational implementation in R language to perform the main functions of the GGE and classical biplot methods; and (ii) to illustrate it by using real data related to genotype-by-environment interaction. This implementation is incorporated by an R package named GGEBiplotGUI, which is a free alternative to the commercial software created by Yan and Tinker (2006) and that can be downloaded from http://cran.r-project.org. The GGEBiplotGUI package is a graphical interface that allows us to use it without necessity of having a knowledge of the R language, which can do this package very attractive for diverse practitioners. The different functions of this package are organized by means of menu options that allows them to be used in a friendly and easy way. Although the main objective of the GGEBiplotGUI package is to construct and manipulate GGE biplots, it is also possible to construct by this package AMMI biplots, only selecting the correct model, as well as the classical biplot. In addition, the GGEBiplotGUI package can be used to analyze genotype-by-environment data because it includes: (i) mega-environment analysis; (ii) test-environment evaluation; and (iii) genotype evaluation. In order to illustrate the capabilities and features of the GGEBiplotGUI package, we use the proposed implementation to analyze real agricultural and genomic data.

The paper is organized as follows. In Sect. 2, we provide the theoretical background of this work. In Sect. 3, we propose and discuss the new computational implementation in R language for biplot methods. In Sect. 4, we perform the empirical application of the proposed computational implementation by using real data. Finally, in Sect. 5, we provide some conclusions.

2 Background

In this section, we provide an overview on classical and GGE biplot methods.

2.1 Classical biplots

Any $n \times m$ matrix $\boldsymbol{Y} = (y_{ij})$ of rank r can be factorized into an $n \times r$ matrix $\boldsymbol{G}$ and an $m \times r$ matrix $\boldsymbol{H}$, both necessarily of rank r, that is, as

$$\boldsymbol{Y} = \boldsymbol{G}\boldsymbol{H}^{\top}. \tag{1}$$

Factorization given in Eq. (1) can be written in a scalar way as

$$y_{ij} = \boldsymbol{g}_i^{\top}\boldsymbol{h}_j, \tag{2}$$

for each i and j, where y_{ij} is the element in the ith row and jth column of $\boldsymbol{Y}$, $\boldsymbol{g}_i^{\top}$ is the ith row of $\boldsymbol{G}$ and $\boldsymbol{h}_j$ is the jth row of $\boldsymbol{H}$. Factorization given in Eq. (2) assigns each vector $\boldsymbol{g}_1, \ldots, \boldsymbol{g}_n$ (row effects) to each one of the n rows of $\boldsymbol{Y}$, and each vector $\boldsymbol{h}_1, \ldots, \boldsymbol{h}_m$ (column effects) to each column of $\boldsymbol{Y}$. Thus, Eq. (2) provides a representation of $\boldsymbol{Y}$ by means $n + m$ vectors in the r-dimensional space. For a matrix of rank equal to two, these $n + m$ vectors may be plotted in the plane, giving a representation of the $n\,m$ elements of $\boldsymbol{Y}$, by means of the inner product of the corresponding row effect and column effect vectors. Such a plot is referred to as a biplot, since it allows row effects and column effects to be jointly plotted; for more details, see Gabriel (1971).

Matrices with ranks greater than two cannot be represented exactly by a biplot. However, if a matrix $\boldsymbol{Y}$ can be approximated by a matrix of rank equal to two, say $\boldsymbol{Y}_{(2)}$, the inner products of the row and column effects can approximate the elements of $\boldsymbol{Y}$.

To approximate any $n \times m$ matrix $\boldsymbol{Y}$ of rank r by an $n \times m$ matrix of smaller rank, one may use the SVD, which can, in this case, be written as

$$\boldsymbol{Y} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{V}^{\top} = \sum_{k=1}^{r} \lambda_k \boldsymbol{u}_k \boldsymbol{v}_k^{\top}, \tag{3}$$

where $\boldsymbol{U} = (\boldsymbol{u}_1, \ldots, \boldsymbol{u}_r)$, $\boldsymbol{V} = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_r)$ and $\boldsymbol{\Lambda} = (\lambda_1, \ldots, \lambda_r)$, with vectors $\boldsymbol{u}_k$ and $\boldsymbol{v}_k^{\top}$ being called left-singular and right-singular vectors, respectively, and λ_k being the singular values. For each $k = 1, \ldots, r$, the singular value λ_k and the singular vectors $\boldsymbol{u}_k$ and $\boldsymbol{v}_k$ are chosen to satisfy

$$\begin{aligned}
\boldsymbol{u}_k^{\top}\boldsymbol{Y} &= \lambda_k \boldsymbol{v}_k \\
\boldsymbol{Y}\boldsymbol{v}_k &= \lambda_k \boldsymbol{u}_k \\
\boldsymbol{Y}\boldsymbol{Y}^{\top}\boldsymbol{u}_k &= \lambda_k^2 \boldsymbol{u}_k \\
\boldsymbol{Y}^{\top}\boldsymbol{Y}\boldsymbol{v}_k &= \lambda_k^2 \boldsymbol{v}_k,
\end{aligned}$$

with $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_r$, such that $\boldsymbol{u}_k^{\top}\boldsymbol{u}_l = \boldsymbol{v}_k^{\top}\boldsymbol{v}_l = \delta_{kl}$, where δ_{kl} denotes the Kronecker delta. Note that, by using the LS method, expression given in Eq. (3) can be written as

$$\boldsymbol{Y}_{(s)n\times m} = \boldsymbol{U}_{(s)n\times s}\boldsymbol{\Lambda}_{(s)s\times s}\boldsymbol{V}_{(s)s\times m}^{\top} = \sum_{k=1}^{s} \lambda_k \boldsymbol{u}_k \boldsymbol{v}_k^{\top}, \tag{4}$$

which corresponds to the best approximation for $\boldsymbol{Y}$ of rank equal to s (see Householder and Young 1938), that is,

expression in Eq. (4) provides an $n \times m$ matrix, say $\boldsymbol{M} = (m_{ij})$, of rank s that minimizes $\| \boldsymbol{Y} - \boldsymbol{M} \|^2 = \sum_{i=1}^{n} \sum_{j=1}^{m} (y_{ij} - m_{ij})^2$. Because $\| \boldsymbol{Y} \|^2 = \lambda_1^2 + \cdots + \lambda_r^2$, the goodness of fit of the model to the data can be measured by $(\sum_{k=1}^{s} \lambda_k^2 / \sum_{k=1}^{r} \lambda_k^2)$ 100 %.

It is also possible to obtain the marker matrices $\boldsymbol{G}$ and $\boldsymbol{H}$ by means of alternating regressions. Specifically, if we consider the row markers $\boldsymbol{G}$ as fixed, the column markers can be computed by regression as (i) $\boldsymbol{H}^\top = (\boldsymbol{G}^\top \boldsymbol{G})^{-1} \boldsymbol{G}^\top \boldsymbol{Y}$. In the same way, fixing $\boldsymbol{H}$, the row markers can be obtained as (ii) $\boldsymbol{G}^\top = (\boldsymbol{H}^\top \boldsymbol{H})^{-1} \boldsymbol{H}^\top \boldsymbol{Y}$. Thus, alternating steps (i) and (ii), their products converge to the SVD; for details about alternating regressions, see Ukkelberg and Borgen (1993).

By choosing factors $\boldsymbol{G}$ and $\boldsymbol{H}$ of $\boldsymbol{Y}_{(2)}$, as in Eq. (1), for biplotting, one may use the factorization provided by the SVD given by $\boldsymbol{G} = \boldsymbol{U}\boldsymbol{\Lambda}^{\gamma}$ and $\boldsymbol{H} = \boldsymbol{V}\boldsymbol{\Lambda}^{1-\gamma}$, where the scalar γ can take any value between zero and one. When $\gamma = 1$, the singular values are entirely partitioned into the row eigenvectors, which is referred to as row-metric preserving or JK biplot; see Gabriel (1971). Because in this case $\boldsymbol{G} = \boldsymbol{U}\boldsymbol{\Lambda}$, we have $\boldsymbol{Y}\boldsymbol{Y}^\top = \boldsymbol{G}\boldsymbol{G}^\top$. Therefore, this partitioning recovers the Euclidean distances among row factors and is, consequently, appropriate for visualizing the similarity/dissimilarity among row factors. When $\gamma = 0$, the singular values are entirely partitioned into the column eigenvectors, referred to as column-metric preserving or GH biplot; see Gabriel (1971). Because now $\boldsymbol{H} = \boldsymbol{V}\boldsymbol{\Lambda}$, we have $\boldsymbol{Y}^\top \boldsymbol{Y} = \boldsymbol{H}\boldsymbol{H}^\top$, which is the sum of squares and cross products matrix of $\boldsymbol{Y}$. If $\boldsymbol{Y}$ is column-centered, then $\boldsymbol{Y}^\top \boldsymbol{Y}$ is $(m - 1)$ times the covariance matrix. Thus, this partitioning is appropriate for studying the relationships among column factors. Two important rules applying to such a partitioning are (i) the correlation between two columns is approximated by the cosine of the angle between their vectors, if the data are column-centered before subjecting to SVD; and (ii) the vector length of a column equals $\sqrt{m-1}$ times the standard deviation (SD) of the column factor, across the rows. When $\gamma = 0.5$, symmetric roles are assigned to rows and columns, referred to as SQRT biplot or symmetric biplot; see Cárdenas and Galindo (2003).

2.2 The HJ biplot

Galindo (1986) proposed a new form of representation, known as HJ biplot, in which the coordinates for columns coincide with the column markers in the GH biplot, whereas the coordinates for the rows coincide with the row markers in the JK biplot. Note that these coordinates may be represented in the same Cartesian system.

The HJ biplot is conceptually similar to the correspondence analysis, but it applies to continuous data rather than to categorical data. The rules for the interpretation of the HJ biplot are a combination of the rules used in classical biplots, correspondence analysis, factor analysis and multidimensional scaling techniques. Specifically, we have that:

(i) the distances among row markers are interpreted as an inverse function of similarities, in a such a way that closer markers (individuals) are more similar; this property allows the clusters of individuals with similar profiles to be identified;
(ii) the lengths of the column markers (vectors) approximate the SD of the variables;
(iii) the cosines of the angles among the column vectors approximate the correlations among variables in a such a way that small acute angles are associated with variables that have high positive correlations, obtuse angles near to the straight angle are associated with variables that have high negative correlations, and right angles are associated with non-correlated variables; in the same way, the cosines of the angles among the variable markers and the axes (principal components) approximate the correlations between them; whereas for standardized data, they approximate the factor loadings in factor analysis; and
(iv) the order of the orthogonal projections of the row markers (points) onto a column marker (vector) approximates the order of the row elements (values) in that column; so, as the projection of a point (individual) away from the center of gravity (average coordinate point), the value that this individual takes on the variable is farther from its mean.

2.3 The GGE biplot

GEI is commonly observed by crop producers and breeders as the differential ranking of cultivar yields among locations or years; see Samonte et al. (2005). Plant breeders conduct MET primarily to identify the superior cultivar for a target region and secondarily to determine whether the target region can be subdivided into different mega-environments; see Yan et al. (2000). Because the main objective of a breeding program is to select genotypes that are consistent and highly productive across different environments, the existence of GEI presents a problem for breeders.

In practice, standard statistical methods that have been applied for analysis of GEI include:

(i) ANOVA model;
(ii) completely multiplicative model (COMM);
(iii) shifted multiplicative model (SHMM);
(iv) genotypes (cultivars) regression model (GREG);
(v) sites (environments) regression model (SREG); and

(vi) AMMI model.

On the one hand, ANOVA is an additive model that describes the main effects and determines whether GEI is a significant source of variation or not. However, it does not provide insight into the genotypes or environments that give rise to the interaction; see details in Samonte et al. (2005). Specifically, for the data matrix $\boldsymbol{Y} = (y_{ij})$, with response variables Y_{ij}, the ANOVA model is

$$Y_{ij} = \mu + \alpha_i + \beta_j + \phi_{ij} + \varepsilon_{ij}, \tag{5}$$

where Y_{ij} is the yield of the genotype i in the environment j, μ is the overall mean, α_i is the genotype (row) main effect, β_j is the environment (column) main effect, ϕ_{ij} is the specific genotype i (row) by the environment j (column) interaction, and ε_{ij} is the error term of the model, where $\varepsilon_{ij} \overset{\text{iid}}{\sim} \text{N}(0, \sigma^2)$. On the other hand, COMM, SHMM, GREG, SREG and AMMI are multiplicative model forms; see Cornelius et al. (1996). Specifically, the COMM is

$$Y_{ij} = \sum_{k=1}^{t} \lambda_k \xi_{ik} \eta_{jk} + \varepsilon_{ij}; \tag{6}$$

the SHMM is

$$Y_{ij} = \mu + \sum_{k=1}^{t} \lambda_k \xi_{ik} \eta_{jk} + \varepsilon_{ij}; \tag{7}$$

the GREG is

$$Y_{ij} = \alpha_i + \sum_{k=1}^{t} \lambda_k \xi_{ik} \eta_{jk} + \varepsilon_{ij}; \tag{8}$$

the SREG is

$$Y_{ij} = \beta_j + \sum_{k=1}^{t} \lambda_k \xi_{ik} \eta_{jk} + \varepsilon_{ij}; \tag{9}$$

and the AMMI model is

$$Y_{ij} = \mu + \alpha_i + \beta_j + \sum_{k=1}^{t} \lambda_k \xi_{ik} \eta_{jk} + \varepsilon_{ij}; \tag{10}$$

where t is the number of SVD axes retained in the model, λ_k is the singular value for the SVD axis k, ξ_{ik} is the singular value of the genotype i for the SVD axis k, η_{jk} is the singular value of the environment j for the SVD axis k, and ε_{ij} is the error term of the models, where $\varepsilon_{ij} \overset{\text{iid}}{\sim} \text{N}(0, \sigma^2)$.

In all of the models in Eqs. (6), (7), (8), (9) and (10), the scale parameters λ_k are ordered as $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_t \geq 0$, and the ξ_{ik} and η_{jk} satisfy the normalization and orthogonality constraints, given by $\sum_i \xi_{ik}^2 = \sum_j \eta_{jk}^2 = 1$ and $\sum_i \xi_{ik} \eta_{im} = \sum_j \xi_{jk} \eta_{jm} = 0$, respectively, for $k \neq m$.

The COMM was used by Fisher and Mackenzie (1923), whereas the SHMM was the first linear-bilinear model used for identifying subsets of genotypes or environments; see Cornelius et al. (1992) and Crossa and Cornelius (1997). The GREG is a reparameterization of the stability analysis model of Finlay and Wilkinson (1963) and Eberhart and Russell (1969). This model expresses the performance of each genotype as its multiple regression on t unknown functions η_{jk} of unidentified characteristics of the sites; see Cornelius et al. (1996). In this context, we use the name "genotype regression" instead of "row regression", mentioned in Bradu and Gabriel (1978). Similarly, we use the name "site regression" instead of "column regression". The multiplicative terms in the models given in Eqs. (6)–(10) are estimated by the LS method through the SVD of the two-way array of data (matrix $\boldsymbol{Z}$), obtained by subtracting the LS estimates from the additive effects of the model, using the cell means; see Cornelius et al. (1996). For these five models, the LS estimates of the additive effects and expressions for the elements of the appropriate $\boldsymbol{Z} = (z_{ij})$ matrix are: [COMM] $z_{ij} = y_{ij}$; [SHMM] $z_{ij} = y_{ij} - \widehat{\mu}$, with $\widehat{\mu} = \overline{y}_{..}$; [GREG] $z_{ij} = y_{ij} - \widehat{\alpha}_i$, with $\widehat{\alpha}_i = \overline{y}_{i.}$; [SREG] $z_{ij} = y_{ij} - \widehat{\beta}_j$, with $\widehat{\beta}_j = \overline{y}_{.j}$; and [AMMI] $z_{ij} = y_{ij} - \widehat{\alpha}_i - \widehat{\beta}_j - \widehat{\mu}$, with $\widehat{\alpha}_i = \overline{y}_{i.} - \widehat{\mu}$, $\widehat{\beta}_j = \overline{y}_{.j} - \widehat{\mu}$, and $\widehat{\mu} = \overline{y}_{..}$ The AMMI model first applies the ANOVA model and then applies the PCA model to the interaction; see Gauch (1988). Then, the genotypic and environmental scores can be used to construct biplots that help to interpret the GEI. GGE applies SVD to the environment-centered two-way data containing G and GE, but the AMMI model applies the SVD to the doubly-centered two-way data, containing GE only. Hence, from Eq. (5), the GGE model is

$$Y_{ij} - \mu - \beta_j = \alpha_i + \phi_{ij} + \varepsilon_{ij}.$$

Thus, the GGE method mixes G and GE and partitions this mixture into t multiplicative terms by

$$Y_{ij} - \mu - \beta_j = \sum_{k=1}^{t} \lambda_k \xi_{ik} \eta_{jk}.$$

Note that the GGE biplot is based on the linear-bilinear SREG; see Yan and Kang (2003). As mentioned, a biplot is as a scatterplot that graphically displays a point or score for each genotype and each environment. To distinguish the members of a model family, the number of components must be considered; for example, a GGE model having two principal components (GGE2) and a AMMI model having two principal components (AMMI2). Two kinds of biplots are common in the yield trial literature: (i) a GGE2 biplot has the first component (PC1) for its abscissa and the second component (PC2) for its ordinate; and (ii) an AMMI2 biplot has also PC1 for its abscissa and PC2 for its ordinate. The interpretation of GGE2 and AMMI2 biplots is similar. Genotypes that are more similar are closer in the plot than genotypes that are less similar. The same is true for environments. Genotypes/environments that are alike

tend to cluster together. The angle between environmental axes is related to the correlation between environments. An acute angle indicates positive correlation, whereas a right angle indicates no correlation, but an obtuse angle indicates negative correlation. The projection of a genotype onto an environmental axis reflects the performance of that genotype in that environment.

3 The `GGEBiplotGUI` package

In this section, we describe the `R` functions (commands) of the `GGEBiplotGUI` package. In this package, three-dimensional biplots are incorporated via the `rgl` package. We recall that the user requires almost no knowledge of the `R` programming language for using our package. The mentioned commands are organized under the following menu entries:

File ⇒ **View** ⇒ **Biplot tools** ⇒ **Format** ⇒ **Models** ⇒ **Biplot**.

Below, we illustrate these entries, describing only those that are not directly visible.

After the `R` program has been downloaded from http://www.r-project.org and installed, it is also necessary to install the `GGEBiplotGUI` package, which depends on the `rgl`, `tcltk` and `tkrplot` packages, which are automatically installed. Then, to load the `GGEBiplotGUI` package into the `R` software, the command

```
library("GGEBiplotGUI")
```

must be entered at the `R` prompt or at any editor program that the user is considering. Once all these instructions are ready, the data, say, for example, `data1`, must be loaded; see Sect. 4. Hence, one produces the corresponding biplot by the command

```
GGEBiplot(data1)
```

An emerging windows, entitled "Model selection", appears, from which one must select the model by choosing the type of SVD among:

1. "JK" (row metric preserving);
2. "GH" (column metric preserving);
3. "HJ" (dual metric preserving); and
4. "SQ" (symmetrical);

the type of centering within:

1. "0" (no centering);
2. "1" (global centering: E + G + GE);
3. "2" (tester centering: G + GE); and
4. "3" (double centering: GE);

and, finally, the type of scaling between:

1. "0" (no scaling); and
2. "1" (std deviation).

After selecting the model, one must click on the OK button and thus the GGE biplot of `data1` will be visualized by an interactive window; for more details about these commands, see Sect. 4 for an example with real data. In this window, a menu with the following options are available:

File

- Open log file.
- Copy image: it copies the current image to the clipboard.
- Save image: it saves the current image in pdf, postscript, metafile, bmp, png, jpg/jpeg or eps/ps (useful for processing lateX files) formats.
- Print image: it prints the current image to any printer connected to the user's computer.
- Exit: it exits the program.

View

- Show both: it shows both entries and testers.
- Show genotypes: it shows genotypes only.
- Show environments: it shows environments only.
- Show/hide title.
- Show/hide guidelines.
- Add/remove symbols: it adds a dot at the location where genotypes are placed.

Biplot tools

- Examine a genotype: it allows the user to select a genotype. When a genotype is selected and the OK button is clicked, the performance of different environments in this genotype is displayed.
- Examine an environment: it allows the user to select an environment. When an environment is selected and the OK button is clicked, the performance of different genotypes in this environment is displayed.
- Relation among environments: it allows us to analyze variability and similarity/dissimilarity among environments in relation to the yield provided by the genotypes.
- Compare two genotypes: it allows a visual comparison of two genotypes with regard to each of the environments. When this function is chosen, a selection panel appears, which allows one to select two genotypes to be compared.
- Which won where/what: it allows an identification of the best genotypes in each environment.
- Discriminativeness against representativeness: it defines an average environment and draws an average-environment-axis (AEA). It allows the representativeness and discriminating power of environments to be visualized.
- Mean against stability: it defines an average environment and draws an AEA. It facilitates visualization of the mean performance and stability of a cultivar.
- Rank environments with ref. to the ideal environment: it defines an ideal environment and compares all

environments to it. The ideal environment is defined as the most discriminating and absolutely representative. It generates a ranking of the test environments in terms of both criteria.
- Rank genotypes with ref. to the ideal genotype: it defines an ideal genotype and compares all genotypes to it. The ideal genotype is defined as one that has the highest performance in all environments and is, therefore, absolutely stable. It generates a ranking of the cultivars in terms of both mean performance and stability.
- Back to original data: it resets the biplot based on the original parameters.

Format

- Biplot title: it allows the biplot title to be modified.
- Change color with the options: background, genotypes labels, environment labels, and biplot title.
- Change font with the options: default, larger, and smaller.

Models

This option allows us to modify the entries initially chosen about "Model selection", which we now detail:

- Scaled (divided by):
  - no scaling;
  - std deviation;
- Centered by:
  - no centering (it corresponds to the model "COMM");
  - global-centered E + G + GE (it corresponds to the model "SHMM");
  - tester-centered G + GE (it corresponds to the model "GGE");
  - double-centered GE (it corresponds to the model "AMMI");
- SVP (singular value partitioning):
  - JK (the singular value partitioning method is "row metric preserving");
  - GH (the singular value partitioning method is "column metric preserving");
  - HJ (the singular value partitioning method is "dual metric preserving");
  - SQ (the singular value partitioning method is "symmetrical");

Biplot

- PC1 against PC2 (default): it shows the first principal component (PC1) against the second principal component (PC2).
- PC3 against PC4: it shows the third principal component (PC3) against the fourth principal component (PC4).
- PC5 against PC6: it shows the fifth principal component (PC5) against the sixth principal component (PC6).
- PC1 against PC3: it shows the first principal component (PC1) against the third principal component (PC3).
- PC2 against PC3: it shows the second principal component (PC2) against the third principal component (PC3).
- Biplot3D: it shows the three dimensional biplot.

4 Data analysis

In this section, we illustrate the `GGEBiplotGUI` package by using real agricultural data. These data correspond to the yield from the 1993 Ontario winter wheat (*Triticum aestivum* L.) performance trials, in which 18 cultivars were tested at nine locations; see Yan and Kang (2003). The data, that we name 'ontario' data (or simply `ontario`), are displayed in Table 1.

Data may be entered using `R` commander or be imported into `R` and saved as a matrix or a data frame. Ontario data have been included in the package as a data frame.

To initialize the GUI with `ontario` data, the command:

```
GGEBiplot(Data = Ontario)
```

must be entered, which opens the "Model selection" window; see Fig. 1.

After selecting the model considering the options "GH", "tester-centered G + GE" and "no scaling", one clicks on the OK button and the main window appears; see Fig. 2(left). The GGE biplot contains markers for each of the 18 cultivars in lower case letters, as distinguished from markers for each of the nine environments in upper case letters. The percentages of GGE explained by the two axes are indicated in Fig. 2(left). The `GGEBiplotGUI` package provides options to view the GGE biplot in numerous ways, in order to address most questions a breeder or researcher is likely to ask. This is exemplified below.

4.1 The performance of different cultivars in a given environment

From the **Biplot tools** menu bar, select "Examine an environment". Choose any environment of interest (BH93, in this example) from the list in the combo-box, and then the following features will appear (see Fig. 2, right):

- A line that passes through the plot origin and the selected environment (BH93, in this example), which is referred to as the environment axis.

Table 1 Mean yield of 18 winter wheat varieties (genotype) tested at nine Ontario locations (environment) for ontario data

Genotype	Environment								
	BH93	EA93	HW93	ID93	KE93	NN93	OA93	RN93	WP93
Ann	4.46	4.15	2.85	3.08	5.94	4.45	4.35	4.04	2.67
Ari	4.42	4.77	2.91	3.51	5.70	5.15	4.96	4.39	2.94
Aug	4.67	4.58	3.10	3.46	6.07	5.03	4.73	3.90	2.62
Cas	4.73	4.75	3.38	3.90	6.22	5.34	4.23	4.89	3.45
Del	4.39	4.60	3.51	3.85	5.77	5.42	5.15	4.10	2.83
Dia	5.18	4.48	2.99	3.77	6.58	5.05	3.99	4.27	2.78
Ena	3.38	4.18	2.74	3.16	5.34	4.27	4.16	4.06	2.03
Fun	4.85	4.66	4.43	3.95	5.54	5.83	4.17	5.06	3.57
Ham	5.04	4.74	3.51	3.44	5.96	4.86	4.98	4.51	2.86
Har	5.20	4.66	3.60	3.76	5.94	5.35	3.90	4.45	3.30
Kar	4.29	4.53	2.76	3.42	6.14	5.25	4.86	4.14	3.15
Kat	3.15	3.04	2.39	2.35	4.23	4.26	3.38	4.07	2.10
Luc	4.10	3.88	2.30	3.72	4.56	5.15	2.60	4.96	2.89
M12	3.34	3.85	2.42	2.78	4.63	5.09	3.28	3.92	2.56
Reb	4.38	4.70	3.66	3.59	6.19	5.14	3.93	4.21	2.93
Ron	4.94	4.70	2.95	3.90	6.06	5.33	4.30	4.30	3.03
Rub	3.79	4.97	3.38	3.35	4.77	5.30	4.32	4.86	3.38
Zav	4.24	4.65	3.61	3.91	6.64	4.83	5.01	4.36	3.11

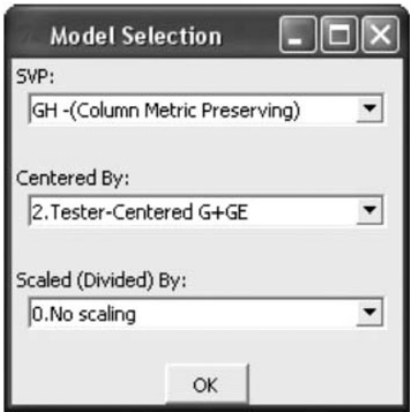


Fig. 1 Window to select the model

- A line that passes through the plot origin and is perpendicular to the environment axis, referred to as the perpendicular line.
- The projections of the cultivars to the environment axis.

Figure 2(right) is based on a "Tester-centered (G + GE)" table, without any scaling and it is row metric preserving (JK biplot). The cultivars are ranked in the direction of the environment axis. In this example cultivar 'fun' was the best, followed by 'cas' and 'har', whereas 'kat' was the poorest in the selected environment BH93. The perpendicular line separates cultivars that performed below average from those performing above average in BH93. Cultivars 'kat', 'm12', 'ena', 'luc', and 'ann' performed below average, whereas other cultivars, on the same side of the perpendicular line as BH93, performed above average.

4.2 The relative adaptation of a given cultivar in different environments

From the **Biplot tools** menu bar, select "Examine a genotype". Choose any genotype of interest (fun, in this example) from the list in the combo-box and then the following features will appear (see Fig. 3, left):

- A line that passes through the plot origin and the select cultivar, which is referred to as the genotype axis.
- A line that passes through the plot origin and is perpendicular to the genotype axis, referred to as the perpendicular line.
- The projections of the environments to the genotype axis.

Figure 3(left) is based on a "Tester-centered (G + GE)" table, without any scaling and it is row metric preserving. The environments are ranked in the direction of the genotype axis in terms of the relative performance of 'fun'. Thus, 'fun' performed the best in WP93, followed by NN93, BH93, RN93, ID93, HW93, EA93, KE93 and OA93. The perpendicular line separates environments, in which 'fun' performed above average from those in which 'fun' performed below average.

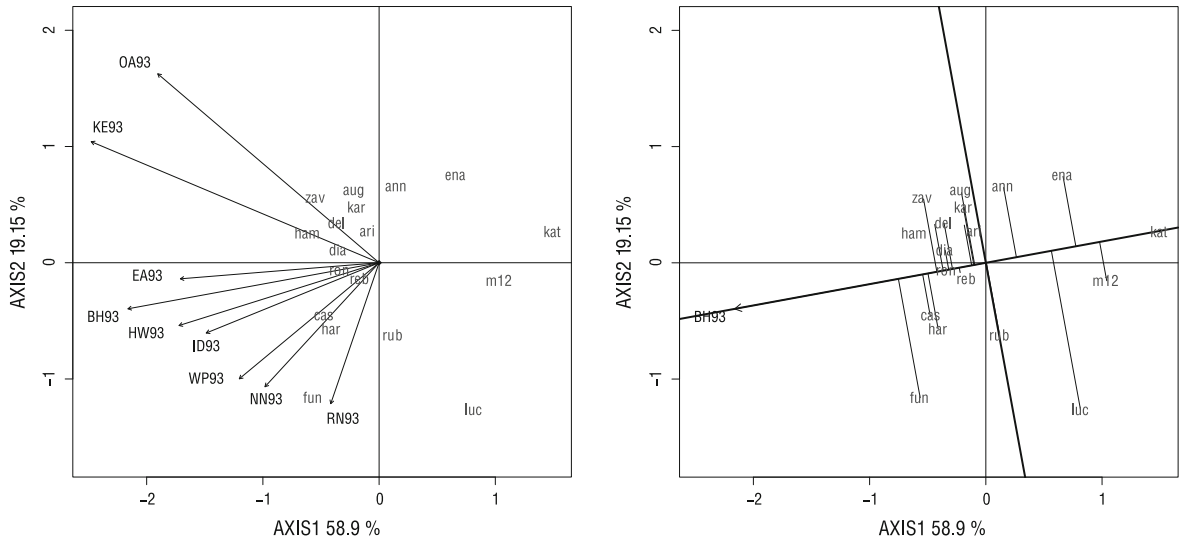


Fig. 2 Main window for ontario data (left) and ranking cultivars based on performance in the environment BH93 for ontario data (right)

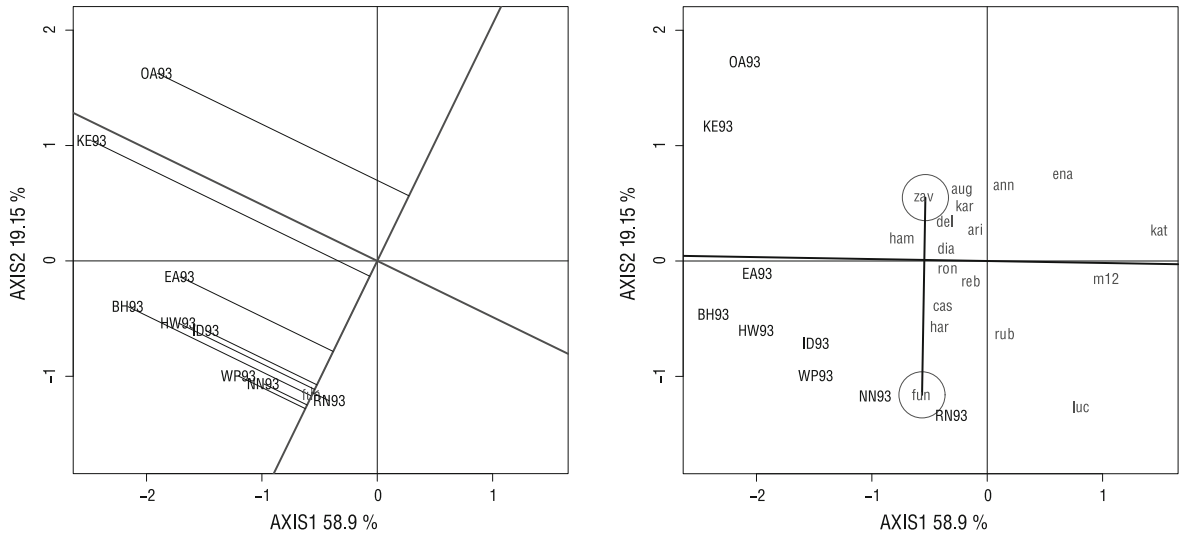


Fig. 3 Ranking environments based on relative performance of a cultivar—fun—for ontario data (left) and comparing 'fun' and 'zav' cultivars for ontario data (right)

4.3 Comparison of two cultivars

From the **Biplot tools** menu bar, select "Compare two genotypes". Choose any cultivar from Genotype 1 (fun), then choose a different cultivar from Genotype 2 (zav), and click the OK button. Upon this clicking, the following features will appear (see Fig. 3, right):

- Two ovals that circle the two selected cultivars.
- A line that connects the two cultivars (jointer line).
- A line that is perpendicular to the jointer line and passes through the plot origin (equality line).

Figure 3(right) is based on a "Tester-centered (G + GE)" table, without any scaling and it is row metric preserving. A cultivar has higher values in environments that are located at its side of the equality line; see Yan and Tinker (2006). In Fig. 3(right), cultivars 'fun' and 'zav' were compared. We see that the environments OA93 and KE93 were on the 'zav' side of the equality line. Thus,

'zav' was better in these environments, but 'fun' was better than 'zav' in the other seven environments.

4.4 Which-won-where

From the **Biplot tools** menu bar, select "Which won where/what". The GGE biplot becomes like Fig. 4(left), which is based on a "Tester-centered (G + GE)" table, without any scaling and it is row metric preserving. The polygon is formed by connecting the markers of the genotypes that are farthest away from the biplot origin, such that all other genotypes are contained in the polygon. Figure 4(left) also contains a set of lines perpendicular to each side of the polygon. These perpendicular lines divide the biplot into several sectors. The winning genotype for each sector is the one located at the respective vertex. Genotypes located at the vertices of the polygon reveal the best or the poorest in one or other environment; see Yan and Tinker (2006). There are five sectors with cultivars 'fun', 'zav', 'ena', 'kat' and 'luc' as the corner or vertex cultivars. Environments OA93 and KE93 fell in the sector in which 'zav' was the vertex cultivar. This means that 'zav' was the best cultivar for OA93 and KE93. The other seven environments fell in the sector in which 'fun' was the vertex cultivar, meaning that 'fun' was the best cultivar for these seven environments. No environments fell into sectors with 'luc', 'ena', and 'kat' as the vertices, indicating that these cultivars were not the best in any of the environments.

4.5 Mean performance and stability of the genotypes

From the **Biplot tools** menu bar, select "Mean against stability". Figure 4(right) is based on a "Tester-centered (G + GE)" table, without any scaling and it is row metric preserving. This figure is the average-environment coordination (AEC) view of the GGE biplot, which has the following interpretation (see Yan and Tinker 2006):

- The single-arrowed line is the AEC abscissa (or AEA) and points to higher mean yield across environments. Thus, 'fun' had the highest mean yield, followed by 'cas', 'har', etc., whereas 'kat' had the lowest mean yield.
- The AEC ordinate passes the plot origin and is perpendicular to the AEC abscissa and points to greater variability (poorer stability) in either direction. Thus, 'rub' was highly unstable, whereas 'cas' was highly stable.

4.6 Ranking genotypes relative to the ideal genotype

From the **Biplot tools** menu bar, select "Rank genotypes with ref. to the ideal genotype". The arrow is where an ideal cultivar should be. Its projection on the AEA was designed to be equal to the longest vector of all cultivars, and its projection on the AEC was obviously zero, meaning that it is absolutely stable. Therefore, genotypes located closer to the ideal genotype are more desirable than others. Thus, 'fun' was more desirable than 'cas', where this last was, of course, the poorest genotype; see Fig. 5(left). This figure is based on a "Tester-centered (G + GE)" table, without any scaling and it is row metric preserving.

4.7 The representativeness and discriminating ability of the environments

Figure 5(right) shows the representativeness and discriminating ability of environments and is based on a "Tester-

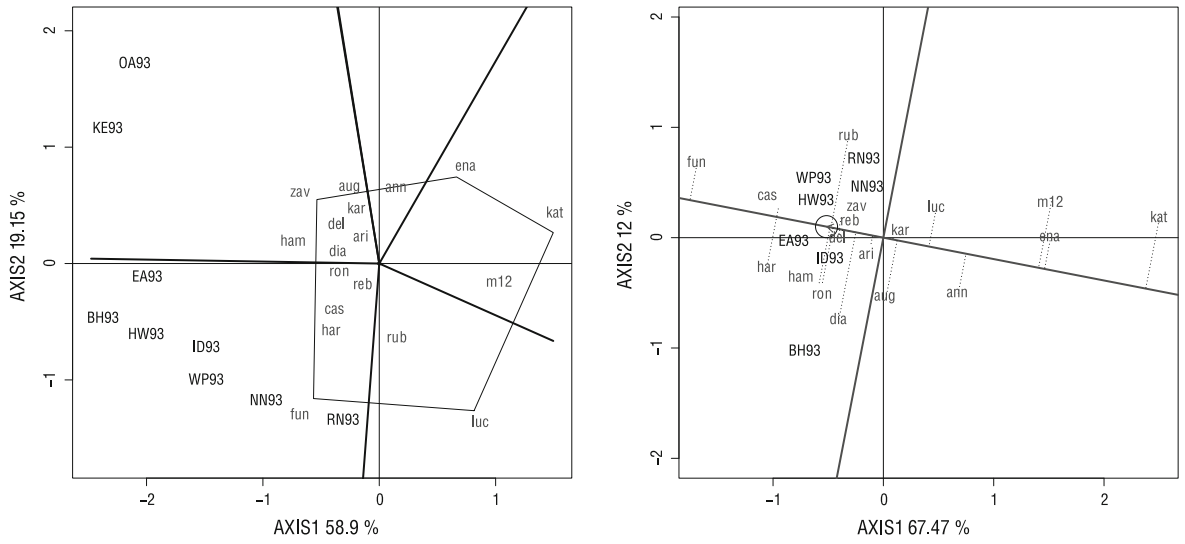


Fig. 4 The which-won-where view of the GGE biplot for ontario data (left) and AEC view for ontario data (right)

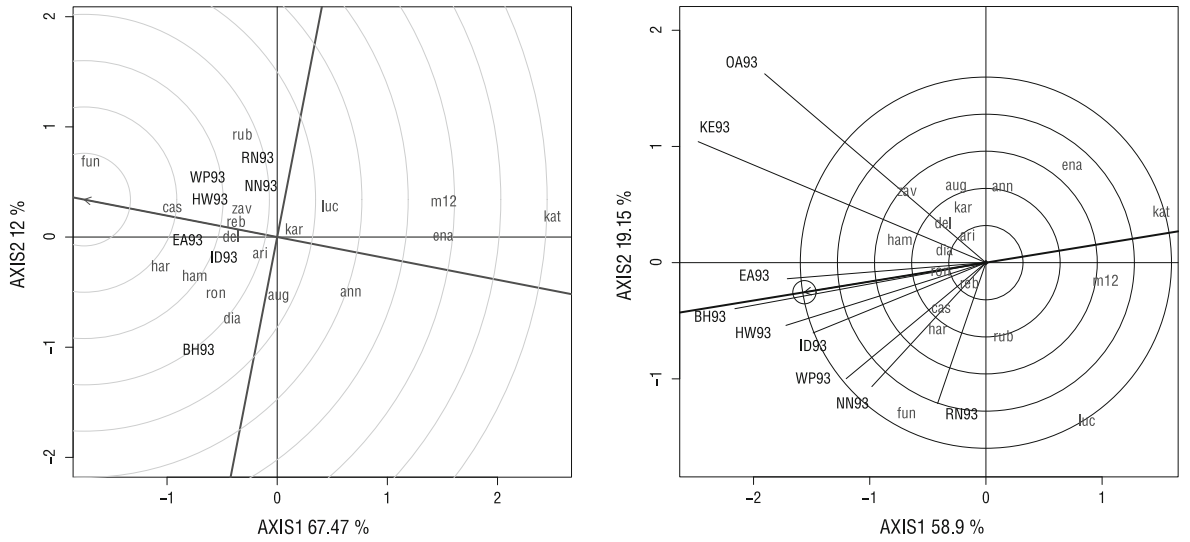


Fig. 5 Ranking cultivars based on both mean performance and stability for ontario data (left) and discriminativeness against representativeness of test environments for ontario data (right)

centered (G + GE)" table, without any scaling and it is column metric preserving (GH biplot). The vector length, that is, the absolute distance between the marker of an environment and the plot origin, is a measure of the discriminating ability: as the longer vector, the discrimination of the environment increases. Therefore, among the nine environments studied, KE93 and OA93 were most discriminating (informative) and RN93 least discriminating. The average environment (represented by the small circle at the end of the arrow) has the average coordinates of all test environments. AEA is the line that passes through the average environment and the biplot origin. A test environment that has a smaller angle with the AEA is more representative of other test environments; see Yan and Tinker (2006). Thus, BH93 is most representative, whereas OA93 is least representative.

4.8 Environment ranking

The ideal test environment should be most discriminating (informative) and also most representative of the target environment. Figure 6 defines an ideal test environment, which is the center of the concentric circles. This is a point on the AEA in the positive direction (most representative), with a distance to the biplot origin equal to the longest vector of all environments (most informative); see Yan and Tinker (2006). BH93 is closest to this point and is, therefore, best, whereas KE93 and OA93 were poorest for selecting cultivars adapted to the whole region. Figure 6 is based on a "Tester-centered (G + GE)" table, without any scaling and it is column metric preserving.

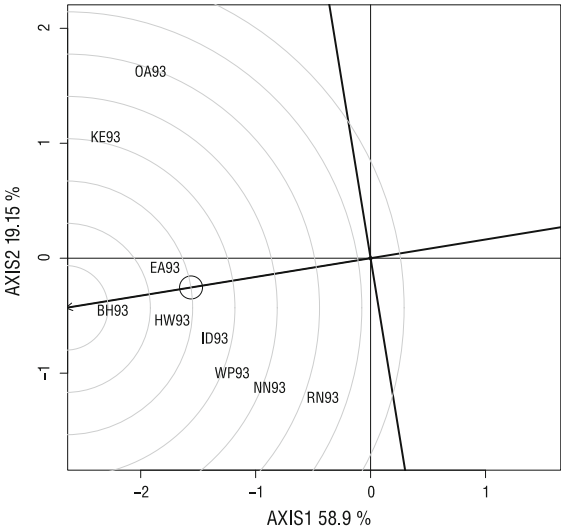


Fig. 6 Ranking environments based on discriminating ability and representativeness for ontario data

5 Conclusions

In this paper, we have proposed a new interactive `R` package called `GGEBiplotGUI`, which has implemented the main functions of the biplot methods. This package is a tool for multi-environment trial data visual analysis and offers a free alternative to the existing commercial software. In addition, the package proposed in this paper is the only open source and interactive software on biplot

methods, with a graphical interface that allows us to use it without necessity of having a knowledge of the R programming language, which can do this package very attractive for diverse practitioners. The GGEBiplotGUI package graphically addresses the main questions that a researcher need likely to ask. Although the biplot method for genotype main effect and genotype by environment interaction was developed originally for analyzing data of multi-environment trials, it can also be used to visualize other types of two-way data. For example, it was satisfactorily employed to visualize diallel cross data (see Yan and Hunt 2002), genotype × trait data (see Yan and Rajcan 2002), and genotype × genetic marker data (see Yan and Falk 2002). The biplot method for genotype main effect and genotype by environment interaction is equally applicable to all types of two-way data that assume an entry × tester structure. Thus, the GGEBiplotGUI package can provide a response to several types of problems, but it mainly is intended to the analysis of data generated by plant breeders and geneticists, in order to visually study yields from genotypes and interactions between genotype and environment. Another interesting feature of this package of free distribution is that it allows us to produce the classical biplots, which can be other attractive aspect for different practitioners.

Acknowledgments The authors wish to thank the Editor-in-Chief, Professor George Christakos, an Associate Editor, and anonymous referees for their comments on an earlier version of this manuscript, which resulted in this improved version. The research of Victor Leiva was partially supported by FONDECYT 1120879 grant from the Chilean government.

References

Bacon-Shone J (2008) Compositional data analysis in the geosciences: from theory to practice (by A. Buccianti, G. Mateu-Figueras and V. Pawlowsky-Glahn, eds). Stoch Environ Res Risk Assess 22:139–141

Barros M, Paula GA, Leiva V (2009) An R implementation for generalized Birnbaum–Saunders distributions. Comput Stat Data Anal 53:1511–1528

Bradu D, Gabriel KR (1978) The biplot as a diagnostic tool for models of two-way tables. Technometrics 20:47–68

Cárdenas O, Galindo MP (2003) Biplot with external information based on generalized bilinear models. Printed by Council of Scientific and Humanistic Development of the Central University of Venezuela, Caracas. Spanish version can be downloaded from bit.ly/14BARON

Choulakian V (1966) Generalized bilinear models. Psychometrika 61:271–283

Cornelius PL, Seyedsadr MS, Crossa J (1992) Using the shifted multiplicative model to search for separability in crop cultivar trials. Theor Appl Genet 84:161–172

Cornelius PL, Crossa J, Seyedsadr MS (1996) Statistical tests and estimators for multiplicative models for genotype-by-environment interaction. In: Kang MS, Gauch HG Jr (eds) Genotype-by-environment interaction. CRC Press, Boca Raton

Crossa J, Cornelius JL (1997) Sites regression and shifted multiplicative model clustering of cultivar trial sites under heterogeneity of error variances. Crop Sci 37:405–415

Denis JB (1991) Ajustements de modelles lineaires et bilineaires sous constraintes lineaires avec donnes manquantes. Stat Appl 39:5–24

Ebdon JS, Gauch HG (2002a) Additive main effect and multiplicative interaction analysis of national turfgrass performance trials I: interpretation of genotype × environment interaction. Crop Sci 42:489–496

Ebdon JS, Gauch HG (2002b) Additive main effect and multiplicative interaction analysis of national turfgrass performance trials II: cultivar recommendations. Crop Sci 42:497–506

Eberhart SA, Russell WA (1969) Yield stability for a 10-line diallel of single-cross and double-cross maize hybrids. Crop Sci 9:357–361

Eckart C, Young G (1939) A principal axis transformation for non-Hermitian matrices. Bull Am Math Soc 45:118–121

Falguerolles A (1995) Generalized bilinear models and generalized biplots: some examples. Publications du Laboratoire de Statistique et Probabilites. Universite Paul Sabatier, Toulouse

Finlay KW, Wilkinson GN (1963) The analysis of adaptation in a plant-breeding programme. Aust J Agric Res 14:742–754

Fisher RA, Mackenzie WA (1923) The manurial response of different potato varieties. J Agric Sci 23:311–320

Gabriel KR (1971) The biplot graphic display of matrices with application to principal component analysis. Biometrika 58:453–467

Gabriel KR (1998) Generalised bilinear regression. Biometrika 85:689–700

Galindo MP (1986) An alternative for simultaneous representation: HJ-biplot. Questíio 10:12–23

Gauch HG (1988) Model selection and validation for yield trials with interaction. Biometrics 44:705–715

Gauch HG (2006) Statistical analysis of yield trials by AMMI and GGE. Crop Sci 46:1488–1500

Gauch HG, Zobel RW (1996) AMMI analysis of yield trials. In: Gauch HG, Kang MS (eds) Genotype-by-environment interaction. CRC Press, Boca Raton, pp 1–40

Gauch GH, Zobel RW (1997) Interpreting mega-environments and targeting genotypes. Crop Sci 37:311–326

Gauch HG, Piepho HP, Annicchiarico P (2008) Statistical analysis of yield trials by AMMI and GGE: further considerations. Crop Sci 48:866–889

Golub GH, Reinsch CH (1970) Singular value decomposition and least squares solution. Numer Math 14:403–420

Gower JC (1992) Generalized biplots. Biometrika 79:475–493

Gower JC, Hand D (1996) Biplots. Chapman & Hall/CRC, London

Gower JC, Harding SA (1988) Nonlinear biplots. Biometrika 75:445–455

Gower JC, Gardner-Lubbe S, Le Roux N (2011) Understanding biplots. Wiley, New York

Greenacre M (2010) Biplots in practice. BBVA Foundation, Madrid

Householder AS, Young G (1938) Matrix approximation and latent roots. Am Math Mon 45:165–171

Kang MS (1988) Using genotype-by-environment interaction for crop cultivar development. Adv Agron 62:199–252

Kempton RA (1984) The use of biplots in interpreting variety by environment interactions. J Agric Sci 103:123–135

Leiva V, Hernandez H, Sanhueza A (2008) An R package for a general class of inverse Gaussian distributions. J Stat Softw 26:1–21

Nelder JA, Wedderburn RWM (1972) Generalized linear models. J R Stat Soc A 135:370–384

R Team (2013) R: a language and environment for statistical computing. R Foundation for Statistical Computing, Vienna. Available at http://www.R-project.org

Samonte SOPB, Wilson LT, McClung AM, Medley JC (2005) Targeting cultivars onto rice growing environments using AMMI and SREG GGE biplot analysis. Crop Sci 45:2414–2424

Ukkelberg A, Borgen O (1993) Outlier detection by robust alternating regressions. Anal Chim Acta 277:489–494

Van Eeuwijk F (1995) Multiplicative interaction in generalized linear models. Biometrics 51:1017–1032

Vicente-Villardón JL, Galindo MP, Blázquez A (2006) Logistic biplots. In: Grenacre M, Blasius J (eds) Multiple correspondence analysis and related methods. Chapman & Hall, New York

Yan W, Falk DE (2002) Biplot analysis of host by pathogen interaction. Plant Dis 86:1396–1401

Yan W, Hunt LA (2002) Biplot analysis of diallel data. Crop Sci 42:21–30

Yan W, Kang MS (2003) GGE biplot analysis: a graphical tool for breeders, geneticists, and agronomists. CRC Press, Boca Raton

Yan W, Kang MS (2006) GGEbiplot. Available at http://www.ggebiplot.com

Yan W, Rajcan I (2002) Biplot evaluation of test sites and trait relations of soybean in Ontario. Crop Sci 42:11–20

Yan W, Tinker NA (2006) Biplot analysis of multi-environment trial data: principles and applications. Can J Plant Sci 86:623–645

Yan W, Hunt LA, Sheng Q, Szlavnics Z (2000) Cultivar evaluation and mega-environment investigation based on GGE biplot. Crop Sci 40:597–605

Yan W, Kang MS, Ma B, Woods S, Cornelius PL (2007) GGE biplot vs. AMMI analysis of genotype-by-environment data. Crop Sci 47:643–655

Data fusion by $T3$-PCA: An integrated global model for the simultaneous analysis of coupled real-valued data

Abstract

In various areas of science, researchers try to get insight into important processes by jointly analysing different data sets containing information regarding common, and possibly also distinctive, aspects of these processes. For example, to explain individual differences in personality, researchers collect for the same set of persons data regarding behavioural signatures (i.e., the reaction profile of a person across different situations), on the one hand, and traits or dispositions, on the other hand. To uncover the processes underlying such coupled data (i.e., two data blocks that have the person mode in common), to all N-way N-mode data blocks simultaneously a global model is fitted in which each data block is represented by a N-way N-mode decomposition model (e.g., PCA, $Parafac$, $Tucker3$) and the parameters underlying the common mode are restricted to be the same for all data blocks this mode belongs to. To estimate the parameters underlying the common mode, an integrated strategy is used that pools the information present in all data blocks (i.e., data fusion). Belonging to the family of component models, one such global model is the recently proposed *Linked-mode Parafac-PCA* ($LMPCA$) model (Wilderjans et al., 2009b,a). For most empirical data, however, $LMPCA$ assumes a too simplistic underlying structure as the three-way data are modelled with the very restrictive $Parafac$ model (i.e., same number of components for each mode and a simple one-to-one correspondence between components of different modes). To overcome these limitations, in this paper, we propose the $T3$-PCA model which represent the three- and two-way data with $Tucker3$ and PCA, respectively. This model is less restrictive than $LMPCA$ because the number of components is allowed to vary across modes and components of different modes may interact in a more complicated way. To estimate the $T3$-PCA model parameters, an alternating least squares algorithm is proposed. The performance of the integrated $T3$-PCA strategy is evaluated and compared with the performance of a segmented strategy (i.e., using only information from one data block to estimate the parameters of the common mode) in extensive simulation study.

Keywords: data fusion, coupled/linked data, integrated strategy, multiblock multiway data analysis, $Tucker3$, PCA, common and distinctive components, segmented strategy, alternating least squares (ALS)

1 Introduction

When investigating psychological and behavioural processes often different types of information are available, with each type of information highlighting common, and possibly also specific, aspects of these processes. For example, to study personality (and individual differences therein), researchers often make use of, on the one hand, behavioural data expressing the reactions of a set of subjects to different situations (i.e., a behavioural signature) and, on the other hand, trait and dispositional information (e.g., scores on openness and neuroticism) or data regarding cognitive-affective units subjects make use of. When each type of information pertains to the same research units, a coupled data set is obtained. In general, a coupled or linked data set can be defined as a set of N-way N-mode data blocks in which each data block has at least one mode in common with at least one other data block (Wilderjans et al., 2009b). In the personality example, the coupled data set consists of a three-way data array (i.e., behavioural signatures) and a two-way trait/disposition data matrix that have one mode (i.e., the subjects) in common.

To disclose the common aspects of the processes underlying the coupled data blocks[1], all the information that is available in the different data blocks should be integrated (i.e., data fusion)[2]. One way to go is to use a (global) model in which the information in each data set is summarized by fitting a N-way N-mode decomposition model (e.g., PCA, $Parafac$, $Tucker3$) to each data block. In a N-way N-mode decomposition model, the data are decomposed into N component matrices, one for each of the N modes of the data, and possibly also a linking array that captures the relations between the components of the N modes (Van Mechelen and Schepers, 2007). For each common mode, the underlying parameters (i.e., common components) are restricted to be the same for all data blocks to which that mode in question belongs to. As a consequence, each common mode is represented by a single set of components.

When fitting such a model to a particular coupled data set, two strategies may be adopted (example of applications of both strategies can be found in ten Berge, 1986; Smilde et al., 2000; Boque and Smilde, 1999; Coxe, 1986). First, a *segmented* strategy in which a (multi-way) decomposition model is fitted to the data block of main interest and, next, the other data block(s) is decomposed with the quantification of the common mode (i.e., common components) being the same as obtained in the first step. Note that in this strategy, different loss functions, one for each data block, are optimized separately. Applied to the personality example, a three-way three-mode decomposition model is fitted to the behavioural signatures data and, in a second step, the disposition data are modelled with a two-way two-mode decomposition model in which the common person components are restricted to equal those from the first analysis. The second strategy, which will be denoted by the term *integrated*, consists of fitting a global model to all data blocks simultaneously. The integrated strategy implies the optimization of a single global loss function, which may be a weighted or unweighted sum of partial loss functions, one for each data block, with the

[1]Note that knowing the common aspects, further enables researchers to detect functional relations between the different data blocks by linking the pieces of information present in each block to each other (i.e., relating situation-specific behavioural profiles to traits).

[2]Different terms are used to denote this type of analysis of coupled data, like *data fusion* (Aerts et al., 2006; Smilde et al., 2005; Van Mechelen and Smilde, 2010), *multiblock data analysis* (Smilde et al., 2003), *analysis of coupled or linked data* (Wilderjans et al., 2009b,a) and *integrative data analysis* (Curran and Hussong, 2009).

weights indicating the extent to which each data block influences the estimation of the common components. Note that these weights may be chosen a priori (e.g., using information on how noisy the data in each data block are, see Wilderjans et al., 2011; van den Berg et al., 2009), or may be determined during the data analysis (e.g., by means of cross-validation, see Smilde and Kiers, 1999). In this paper, it will be assumed that these weights are known at the start of the analysis (see Section 6). As such, when estimating the parameters of the common mode, the information that is present in all data blocks to which this common mode belongs to, is used (i.e., data fusion). For our personality example, the behavioural signature and disposition data blocks are modelled simultaneously with an appropriate decomposition model and the common components, which are equal for both data blocks, are estimated by using the information present in both data blocks.

A segmented strategy may be used when it cannot be assumed that a coupled data set has common components and/or when the structure underlying one (or more) data block is of primary interest (and the other data block-s- is only used to guide the interpretation of the main data block). A disadvantage of a segmented strategy, however, is that the common components are estimated using only information from a single data block, resulting in the noise in that data block having a considerable influence on the obtained solution (also for the structure underlying the other data blocks). An integrated strategy may tackle this disadvantage by pooling all information present in the data blocks when estimating the common components. As such, noise may get filtered out, resulting in more correct and stable inferences regarding the structure underlying the coupled data (for a discussion of advantages and disadvantages of both strategies and a comparison of both strategies by means of an extensive simulation study in the context of hierarchical classes ($HICLAS$) models, see Wilderjans et al., 2008).

In this paper, we will confine out attention to a coupled data set consisting of a real-valued three-way data array that shares a single mode with a real-valued two-way data array (e.g., behavioral signature and dispositional data collected for the same set of subjects). Recently, Wilderjans et al. (2009a) proposed the $linked\text{-}mode\ PARAFAC\text{-}PCA$ ($LMPCA$) model to analyse such coupled data. The $LMPCA$ model is a global model in which the three-way array is decomposed according to $Parafac$ (Hitchcock, 1927; Harshman, 1970; Carroll and Chang, 1970) and the two-way matrix with PCA (Pearson, 1901; Hotelling, 1933; Jolliffe, 2002) and the components for the common mode are constrained to be equal for both data blocks. To estimate the parameters of the $LMPCA$ model, the authors use an integrated fitting strategy in which a weighted loss function is optimized by means of an alternating least squares (ALS) algorithm (ten Berge, 1993; de Leeuw, 1994; Kroonenberg and De Leeuw, 1980). When analysing psychological data, however, $LMPCA$ may not yield a good description of the underlying structure as the $Parafac$ model underlying the three-way data is quite restrictive because it imposes a set of constraints that are not often met for psychological data. In particular, $Parafac$ assumes that the number of components is equal for each mode. Moreover, components from different modes are only related to one another in a one-to-one fashion, implying that the component loadings for the different slices (i.e., third mode of the array) are assumed to be proportional to each other. Applied to our example, there is no theoretical ground to assume that the variation in the persons, responses and situations can be adequately captured by the same number of components. Moreover, it is also often not tenable that the same combinations of person and behaviour components, although they may dif-

fer in strength (i.e., proportionality), play a role in each encountered situation. To address these limitations, in this paper, we will propose the $T3$-PCA model which is a novel global model for an integrated analysis of the type of coupled data we focus on. In this model, the three-way data block is decomposed according to a $Tucker3$ model (Tucker, 1966, 1964) and the two-way data are modelled with PCA with the restriction of the common components to be equal for both data blocks. By decomposing the three-way data block according to a $Tucker3$ model, a less restrictive global model is obtained as $Tucker3$ allows the number of components to differ across modes and implies a linking structure between the components of the reduced modes which captures the interactions between these components.

The remainder of this paper is organized in five main sections. In Section 2, after discussing the structure of the data and options for pre-processing, the novel $T3$-PCA model is presented and the relations with related models are discussed. In the third section, the loss function is introduced and an ALS algorithm to estimate the model parameters is described, together with heuristics for rank selection and post-processing issues. In the following two sections, the performance of the integrated $T3$-PCA strategy is studied and compared with the performance of a segmented strategy. In the fourth section this is done by means of an extensive simulation study. To round off, Section 5 contains some concluding remarks.

2 Model

2.1 Data structure and pre-processing

Data structure. A linked-mode $T3$-PCA analysis requires a (three-way) $I \times J \times K$ object by attribute by source real-valued data array $\underline{\mathbf{D}}^1$ and a (two-way) $I \times L$ object by covariate data matrix $\mathbf{D}^2$ that have the objects (i.e., the elements of the first mode) in common. For example, in personality research, as can be seen in the graphical presentation of such a data structure in Figure 1, the data array $\underline{\mathbf{D}}^1$ may contain for a set of participants their behaviour profile to different situations, whereas $\mathbf{D}^2$ may have the scores for the same set of participants on a list of personality items. Other examples of these type of data (for an overview, see **?**) can be found in psychometrics (Wilderjans et al., 2009b,a), neuropsychology (Mitchell et al., 2008; Palatucci et al., 2009; Chiang et al., 2010), community detection (Lin et al., 2009), collaborative filtering (Zheng et al., 2010; Ermis et al., 2012), bioinformatics (Acar et al., 2012) and chemometrics (Smilde et al., 2000). It is important to notice that the set of elements for the common mode in $\underline{\mathbf{D}}^1$ and $\mathbf{D}^2$ is identical (i.e., the same set of participants in the personality psychology example).

Pre-processing. Pre-processing often involves operations of centering and scaling that aim at removing arbitrarily differences between the variables regarding offsets (i.e., scale means) and scale variances. A classical way of pre-processing a two-way data block (as is done in, for example, PCA) is to center (i.e., mean of zero) and standardize (i.e., variance of one) each variable (Bro and Smilde, 2003). For a three-way data block, however, different centering and scaling options exist (Kroonenberg, 1983, 2008; Smilde et al., 2004). Because we want to disclose the common mechanisms underlying both data blocks, we advise, prior to an actual $T3$-PCA analysis, to re-

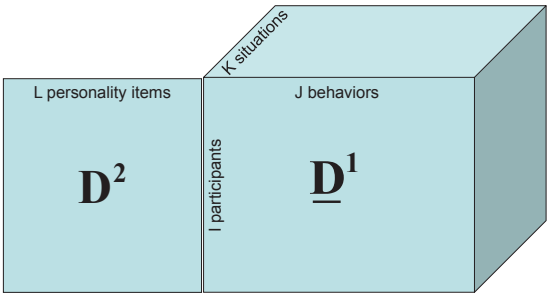


Figure 1: Graphical representation of the data for a linked-mode $T3$-PCA analysis.

move differences between both data blocks in overall variable means. To this end, in the three-way array, the mean of each attribute-source combination (across the objects) is set to zero, whereas in the two-way matrix, each covariate is centered (across the objects). Further, to account for arbitrary differences in scale variances (see Timmerman and Kiers, 2003), in the three-way array the variance of each attribute (computed across objects and sources) is set to one (i.e., sum of squares of IK), whereas in the two-way matrix the variance of each covariate is fixed to one (i.e., sum of squares of I). As a consequence, the importance of each variable, considered both for the whole data set as for each data block separately, is set equal in the analysis.

2.2 A linked-mode model for coupled real valued data: $T3$-PCA

$\underline{Model}$. The linked-mode $T3$-PCA model approximates an $I \times J \times K$ real-valued data array $\underline{\mathbf{D}}^1$ that is coupled/linked with an $I \times L$ real-valued data matrix $\mathbf{D}^2$ through the first mode by a real-valued model array $\underline{\mathbf{M}}^1$ (of size $I \times J \times K$) and a real-valued model matrix $\mathbf{M}^2$ (of size $I \times L$), respectively, that have the first mode in common. In particular, $\underline{\mathbf{D}}^1$ is approximated by an $I \times J \times K$ real-valued model array $\underline{\mathbf{M}}^1$ that can be decomposed according to a $Tucker3$ model (Tucker, 1964, 1966; Kroonenberg and De Leeuw, 1980) with P object, Q attribute and R source components; $\mathbf{D}^2$ is approximated by an $I \times L$ real-valued model matrix $\mathbf{M}^2$ that can be decomposed according to a PCA model with P components, with the component matrix for the first mode (i.e., rows) in both

$\underline{\mathbf{M}}^1$ and $\mathbf{M}^2$ being restricted to be identical. Therefore, $\underline{\mathbf{M}}^1$ and $\mathbf{M}^2$ can be decomposed as follows:

$$m^1_{ijk} = \sum_{p=1}^{P}\sum_{q=1}^{Q}\sum_{r=1}^{R} a_{ip}b^1_{jq}c^1_{kr}g_{pqr} \quad (1)$$

$$m^2_{il} = \sum_{p=1}^{P} a_{ip}b^2_{lp}, \quad (2)$$

where a_{ip}, b^1_{jq}, c^1_{kr} and b^2_{lp} denote the entries of the real-valued component matrices $\mathbf{A}$ $(I \times P)$, $\mathbf{B}_1$ $(J \times Q)$, $\mathbf{C}$ $(K \times R)$, and $\mathbf{B}_2$ $(L \times P)$ for the objects, attributes, sources and covariates, respectively, and P, Q and R denote the number of object, attribute and source components, respectively. The entries g_{pqr} denote the elements of the real-valued core array $\underline{\mathbf{G}}$ (of size $P \times Q \times R$) and reflect the relationship between the p^{th}, q^{th} and r^{th} component in $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$, respectively. Note that $\underline{\mathbf{M}}^1$ and $\mathbf{M}^2$ are linked to each other by restricting the common object component matrix a to be identical in both submodels. It should be noted that the parameters of the linked-mode $T3$-PCA model (i.e., the componentn matrices and the core array) are unique upon a joint permutation, scaling, reflection, and rotation of the components and/or core array (see Smilde et al., 2004). To partially identify the solution, we will constrain $\mathbf{A}$, $\mathbf{B}_1$ and $\mathbf{C}$ to be columnwise orthonormal (i.e., $\mathbf{A}'\mathbf{A}$ = $(\mathbf{B}_1)'\mathbf{B}_1$ = $\mathbf{C}'\mathbf{C} = \boldsymbol{I}$, which implies that components are orthogonal and have unit length)[3].

Relation to other models. The linked-mode $T3$-PCA model belongs to the family of multiway multiblock component models, which subsumes different models like, amongst others, the $linked\text{-}mode\ PARAFAC - PCA$ model ($LMPCA$; Wilderjans et al., 2009b,a). The novel linked-mode $T3$-PCA model is related to several existing models both inside and outside the family of multiway multiblock component models. Inside this family, the linked-mode $T3$-PCA model can be considered as a generalization of the $LMPCA$ model, in the same vein as $Tucker3$ is a generalization of $Parafac$. In particular, when constraining $P = Q = R$ and $\underline{\mathbf{G}}$ to be a superidentity matrix (i.e., $g_{rst} = 1$ if $r = s = t$ and $g_{rst} = 0$ otherwise), the linked-mode $Tucker3 - PCA$ model is equivalent to the $LMPCA$ model. The family of multilevel multiblock component models is further closely related to the family of multiway covariate regression models (also named called multiway multiblock regression models, see Smilde and Kiers, 1999; Smilde et al., 2000). The main difference between both model families pertains to the role that is assigned to the different data blocks (i.e., whether or not blocks have exchangeable roles in terms of conceptual status). In a multiway multiblock component model, all data blocks serve the same role, whereas in a multiway covariates regression model, some blocks are considered predictor blocks that are to be regressed on the other block(s) that functions as criterion block(s).

[3]Note that these constraints, which do not alter the fit of the model, imply that all the variation in the three-way block is carried to the core elements (resulting in the square of each core elements denoting the percentage of explained variance by the specific combination of row, column and slice components, see Kroonenberg, 1983, 2008) and all the information in the two-way block to $\mathbf{B}_2$. As a consequence, in general, $\mathbf{B}_2$ will not be orthogonal nor normalized and restricting $\mathbf{B}_2$ to be orthonormal implies a quite restricted model (a.o., the P non-zero eigenvalues of $\mathbf{M}^2$ being equal to one). It should be remarked that additional restrictions are needed to fully identify the model (amongst others, to deal with the rotational freedom; for more information, see Smilde et al., 2004). This non-uniqueness, however, as is done in PCA, can be used to get a solution that is easier to interpret (see the discussion of post-processing strategies in Section 3.4).

3 Data Analysis

3.1 Aim

The aim of a linked-mode $T3$-PCA analysis with rank (P, Q, R) of a coupled data set ($\underline{\mathbf{D}}^1, \mathbf{D}^2$) is to estimate a coupled model ($\underline{\mathbf{M}}^1, \mathbf{M}^2$) such that the value of the loss function

$$f = \frac{\alpha}{\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}(d_{ijk}^1)^2} \times \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}(d_{ijk}^1 - \sum_{p=1}^{P}\sum_{q=1}^{Q}\sum_{r=1}^{R} a_{ip}b_{jq}^1 c_{kr} g_{pqr})^2 \qquad (3)$$
$$+\frac{(1-\alpha)}{\sum_{i=1}^{I}\sum_{l=1}^{L}(d_{il}^2)^2} \times \sum_{i=1}^{I}\sum_{l=1}^{L}(d_{il}^2 - \sum_{p=1}^{P} a_{ip}b_{lp}^2)^2$$

is minimized (with $0 \leq \alpha \leq 1$), $\underline{\mathbf{M}}^1$ can be represented by a $Tucker3$ model with P, Q and R components (and $\mathbf{A}$, $\mathbf{B}_1$ and $\mathbf{C}$ being orthonormal), $\mathbf{M}^2$ by a PCA model with P components and with $\mathbf{A}$ to be constrained equal in the decomposition of $\underline{\mathbf{D}}^1$ and $\mathbf{D}^2$. The weights α and $1-\alpha$ denote the extent to which each of the two data blocks influence the analysis (Wilderjans et al., 2009a). When α equals 1 or 0, a segmented approach is adopted in which the components of the common mode are derived based on the information in one data block only (i.e., the three-way and two-way data block, respectively). For intermediate values of α, the common components are estimated using the (weighted) information in both data blocks, which can be conceived as an integrated approach. In order to let differences between the scales on which the data blocks are measured not influences the analysis, we normalized the data blocks by dividing the sum of squared differences between data and model for each block by the sum of squared data entries in each block[4]. As, in general, the three-way data block will have a larger sum of squared data entries, this normalizing implies that the influence of the three-way block is downweighted in the analysis. Note that the loss function in (3) can be rewritten as (Vervloet et al., 2013):

$$f_2 = \beta \times \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}(d_{ijk}^1 - \sum_{p=1}^{P}\sum_{q=1}^{Q}\sum_{r=1}^{R} a_{ip}b_{jq}^1 c_{kr} g_{pqr})^2 + (1-\beta) \times \sum_{i=1}^{I}\sum_{l=1}^{L}(d_{il}^2 - \sum_{p=1}^{P} a_{ip}b_{lp}^2)^2 \quad (4)$$

with $\beta = \frac{\alpha \times \sum_{i=1}^{I}\sum_{l=1}^{L}(d_{il}^2)^2}{\alpha \times \sum_{i=1}^{I}\sum_{l=1}^{L}(d_{il}^2)^2 + (1-\alpha) \times \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}(d_{ijk}^1)^2}$ and f_2 in (4) and f in (3) being equal to each other upon a scaling factor that is irrelevant for the optimization (i.e., f and f_2 having the same optimal solution).

3.2 Algorithm

[4]When scale differences (in variance) exist regarding the variables within a data block, one may consider to apply this normalization procedure variablewise instead of blockwise.

To estimate the parameters of the linked-mode $Tucker3$-PCA model, an alternating least squares (ALS) algorithm is used (Kroonenberg and De Leeuw, 1980). In this algorithm, after obtaining starting values for all model parameters, the components matrices $\mathbf{A}$, $\mathbf{B}_1$, $\mathbf{C}$, and $\mathbf{B}_2$ and the core array $\underline{\mathbf{G}}$ are alternatingly re-estimated conditionally upon the other parameters until a prespecified convergence criterion is satisfied (see, ten Berge, 1993).

The initial estimates of the component matrices can be determined either rationally or randomly (for a discussion of different types of starting configurations, see Ceulemans et al., 2007). Rational initial estimates of $\mathbf{A}$, $\mathbf{B}_1$, and $\mathbf{C}$ are obtained by taking the eigenvectors associated with the P, Q and R largest eigenvalues of $[\mathbf{D}_a^1|\mathbf{D}^2][\mathbf{D}_a^1|\mathbf{D}^2]'$, $\mathbf{D}_b^1(\mathbf{D}_b^1)'$ and $\mathbf{D}_c^1(\mathbf{D}_c^1)'$, respectively, where $\mathbf{D}_a^1$, $\mathbf{D}_b^1$ and $\mathbf{D}_c^1$ are the $I \times JK$, $J \times KI$, and $K \times IJ$ matricized versions of $\underline{\mathbf{D}}^1$ and $|$ denotes matrix concatenation (Kiers, 2000). Note that this procedure ensures that the initial $\mathbf{A}$, $\mathbf{B}_1$, and $\mathbf{C}$ are orthonormal. Random initial estimates for the component matrices are obtained by, first, sampling entries from a standard normal distribution and next, in order to get orthonormal initial component matrices, to compute an orthonormal basis for the range of each of the initial component matrices. Once having initial values for $\mathbf{A}$, $\mathbf{B}_1$ and $\mathbf{C}$, initial estimates for $\mathbf{B}_2$ and $\underline{\mathbf{G}}$ can easily be obtained by means of linear regression (see further).

From initial estimates of the model parameters, the algorithm performs a number of iterations, until a specified stop criterion is met. In each iteration, the component matrices and the core array are re-estimated conditionally on the other parameters. In particular, first, fixing the entries of $\mathbf{B}_1$, $\mathbf{C}$ and $\mathbf{B}_2$, the entries of $\mathbf{A}$ and re-estimated. Next, the entries of $\mathbf{B}_1$ (conditionally upon $\mathbf{A}$ and $\mathbf{C}$) and $\mathbf{C}$ (conditionally upon $\mathbf{A}$ and $\mathbf{B}_1$) are updated. Finally, $\underline{\mathbf{G}}$ (conditionally upon $\mathbf{A}$, $\mathbf{B}_1$ and $\mathbf{C}$) and $\mathbf{B}_2$ (conditionally upon $\mathbf{A}$) are re-estimated. The conditional updating of the component matrices $\mathbf{B}_1$, $\mathbf{C}$, $\mathbf{B}_2$ and the core matrix $\underline{\mathbf{G}}$ boils down to a (constrained) multivariate linear regression problem, which has a closed form solution (see Kroonenberg and De Leeuw, 1980; Smilde et al., 2004; Andersson and Bro, 1998; Bro and Andersson, 1998). Finally, $\mathbf{A}$, which is shared by the $Tucker3$ model for the three-way data and the PCA model for the two-way data, can be re-estimated by $\mathbf{A} = \mathbf{U}\mathbf{V}'$ where $\mathbf{U}$ and $\mathbf{V}$ are the P left and right singular vectors, respectively, associated with the P largest singular values of $\mathbf{Y}\mathbf{Z}'$, with $\mathbf{Y} = [\sqrt{\alpha} \times \mathbf{D}_a^1 \mid \sqrt{(1-\alpha)} \times \mathbf{D}^2]$ and $\mathbf{Z} = [\sqrt{\alpha} \times \mathbf{G}_a\,(\mathbf{C} \otimes \mathbf{B}_1)' \mid \sqrt{1-\alpha} \times \mathbf{B}_2']$. After each iteration, the loss function value is re-computed, using the updated parameter estimates. When updating the component matrices and the core involves a substantial decrease in the loss function (i.e., a decrease larger than or equal to a certain proportion - e.g., .0000001 - of the loss function value of the previous iteration), a new iteration is performed; otherwise the algorithm stops, since it has converged, and the current estimates of the model parameters are retained as the final estimates.

As updating the model parameters during the iterative process never can lead to an increase in the loss function value and the loss function is bounded below by zero, the $T3$-PCA algorithm is guaranteed to converge. This, however, does not imply that the algorithm always will identify the global optimal parameter estimates, as the loss function may suffer from the presence of (many) locally optimal solutions. To minimize the risk of the $T3$-PCA algorithm ending up in a local optimal solution, it is advised to adopt a multistart procedure. In such a procedure, the algorithm is run multiple times, each time starting with different randomly or pseudo-randomly[5] obtained

[5]Pseudo-random initial values can be obtained by slightly perturbating a rationally obtained starting solution (for

initial parameter values, and the solution from the run yielding the lowest loss function value is retained.

3.3 Model Selection

In general, no a priori information is available regarding the optimal rank (P,Q,R) underlying a $T3$-PCA model for a data set at hand. As a way out, researchers may perform different analysis with varying complexities in which the rank of the model goes from (1,1,1) to (P,Q,R)[6], and select a model that has a good balance between model fit and model complexity; model (mis)fit may be quantified by the optimal loss function value, whereas different options exist to define model complexity, like the total number of components (i.e., $P + Q + R$), the total number of fitted parameters (i.e., $IP + JQ + KR + PQR + LP$) and the number of free parameters (i.e., $IP + JQ + KR + PQR + LP - P^2 - Q^2 - R^2$, see Ceulemans and Kiers, 2006; Kroonenberg and ten Berge, 2011). Next, the optimal rank of a $T3$-PCA model may be identified by using the CHull procedure (Ceulemans and Kiers, 2006; **?**; Wilderjans et al., 2013), which is an automated heuristic for model selection.

Starting from a fit f_i and complexity c_i value for all valid $T3$-PCA solutions (see footnote 6) with a rank between (1,1,1) and (P,Q,R), the CHull method goes through the following two steps: (1) determining the models m_i ($i = 1, ..., M$) that are located on the (lower) boundary of the convex hull of the c_i by f_i complexity-by-fit plot of all the valid models and (2) identifying the model on the boundary of the convex hull that optimally balances model (mis)fit and model complexity. To this end, one may use an automated procedure by computing for each model m_i the corresponding *st*-ratio:

$$st_i = \frac{f_i - f_{i-1}/c_i - c_{i-1}}{f_{i+1} - fi/c_{i+1} - c_i},$$

and selecting the model with the largest *st* value. This solution optimally balances model fit and model complexity as increasing the complexity has only a small effect on the fit measure (i.e., a small denominator), whereas lowering complexity implies a substantial drop in fit (i.e., a large numerator). Note that for the first and last model (i.e., m_1 and m_N) the *st*-ratio is not defined and, as a consequence, these models cannot be selected (for a discussing regarding which model to include in the procedure, see Wilderjans et al., 2013).

3.4 Post-processing and rotational freedom

As both the $Tucker3$ and PCA decomposition of the three-way and two-way data block, respectively, suffer from rotational freedom (even after imposing the orthonormality constraint on $\mathbf{A}$, $\mathbf{B}_1$ and $\mathbf{C}$), also the $T3$-PCA decomposition is not unique (see also Section 3.1). In particular, the

more information, see Ceulemans et al., 2007; Wilderjans and Ceulemans, 2013).

[6]It should be remarked that models for which $Q > PR$ or $R > PQ$ should not be considered as it can be shown that there exists models that are more parsimonious (i.e., have a smaller number of fitted parameters) that fit the data equally well (i.e., have the same loss function value, see Wansbeek and Verhees, 1989; Ceulemans and Kiers, 2006).

component matrices $\mathbf{A}$, $\mathbf{B}_1$ and $\mathbf{C}$ can be independently rotated along as the core array $\underline{\mathbf{G}}$ (and, in case of $\mathbf{A}$, also $\mathbf{B}_2$) is appropriately counterrotated. Moreover, $\mathbf{A}$ or $\mathbf{B}_2$ can be rotated as long as the other matrix and the core array are appropriately counterrotated.

The rotational freedom can be used to obtain a solution that is easier to interpret. To this end, different strategies can be followed: (1) rotate the component matrices (by means, for example, of a varimax rotation in order to obtain a simple structure, see Kaiser, 1958) and afterwards adjust the core matrix, (2) rotate the core matrix and afterwards counterrotate the component matrices (Kiers, 1992, 1997). The first option yields components that are easy to interpret but probably at the expense of a complicated core array containing many large numbers (i.e., complicated relations between the components). Contrarily, the second option results in a parsimonious core (i.e., only a few large values indicating the most important interactions between the components) but components that may be hard to interpret.[7]

4 Simulation Study

4.1 Problem

In this section, we will present a simulation study to evaluate the performance of the $T3$-PCA algorithm, herewith focusing on the following two performance aspects: (1) optimization of the loss function and (2) parameter recovery. Regarding the former, we will study how sensitive the algorithm is to the problem of local minima. Concerning recovery, the simulation study has the following two aims: (1) examining to which extent the $T3$-PCA algorithm uncovers the underlying component structure and (2) determining the degree to which $T3$-PCA, which is a simultaneous strategy (i.e., using the information in both the three-way array and the two-way data matrix to obtain the components -in **A**- for the common mode), outperforms a sequential strategy (i.e., extracting the components for **A** from the three-way data array and next decomposing the two-way data matrix with the restriction that **A** should be the same).

To address the latter aim, it is important to distinguish between components for the common mode that explain a sizeable amount of variance in both data blocks (i.e., common components) and components that are only underlying one of both data blocks (i.e., distinctive components, see Van Deun et al., 2013; Schouteden et al., 2013, 2014). When only common components are underlying the coupled data set, it can be expected that a segmented strategy (i.e., first fitting $Tucker3$ to the three-way data) already performs quite well, leaving (almost) no room for the integrated strategy to perform better. The reason for this is that the $Tucker3$ structure imposed on the three-way data set is quite restrictive, resulting in an adequate estimation of the common component matrix (i.e., the $Tucker3$ algorithm filters out the noise in the data quite well, see Wilderjans et al., 2009b, 2008). However, when the common mode of the three-way data block contains a strong distinctive component (in terms of amount of explained variance) and one or more small common components, a segmented strategy may be tempted to retain the distinctive

[7] Note that after fixing the rotational freedom, the $T3$-PCA model is still only identified upon a permutation and/or reflection of the components. As this non-uniqueness does not have any thorough consequences for the interpretation of the solution, it is not discussed here (for more information, see Smilde et al., 2004).

component at the expense of the small common ones because the information from the two-way block is not used and model selection strategies are aimed at mainly selecting strong components (**?**). As an integrated strategy takes the information in both the three-way and two-way data block into account, we predict that this strategy will be able to ignore the strong but distinctive component and will retain the (less strong) common component(s).

To test our prediction, we will set up a simulation study in which both data blocks contain an extra component (above the common ones) that is distinctive and we will investigate how recovery performance is affected by the strength of this distinctive component. Furthermore, we will also study how the recovery performance depends on the following four (other) data characteristics: (1) the size of the three-way data block, (2) the ratio of the size of the three-way data block to the size of the two-way block, (3) the rank of the model and (4) the amount of noise in the data. Regarding these manipulated factors, we expect that the performance of the $T3$-PCA algorithm will decrease when the data become more complex (i.e., a larger rank and more noise, see van den Berg et al., 2009; Wilderjans et al., 2012b,a; Wilderjans and Ceulemans, 2013) and when the distinctive component becomes stronger. Further, we expect the integrated strategy to outperform the segmented strategy to a larger extent when the distinctive component explains more variance in the data. Finally, we will also explore how recovery performance is related to the choice of α.

4.2 Design and procedure

The factors that were introduced above were systematically manipulated in a completely randomized five-factorial design in which all factors were considered random:

1. The size ($I \times J \times K$) of the true three-way data array $\underline{\mathbf{T}}^1$, at 2 levels: ($20 \times 20 \times 20$) and ($20 \times 20 \times 100$);

2. The ratio *r* of the size of the true three-way data array $\underline{\mathbf{T}}^1$ to the size of the true two-way data matrix $\mathbf{T}^2$:
$$r = \frac{I \times J \times K}{I \times L}.$$
This factor was manipulated at two levels: 1 and 0.25;

3. The rank of the true model ($P \times Q \times R$) at 2 levels: $(2, 2, 2)$ and $(2, 3, 4)$;

4. The amount of noise in the data, ε, at 2 levels: $.15$ and $.30$;

5. The amount of variance that the (extra) distinctive component for the common mode in the two- and three-way data block explains. This factor was manipulated at three levels: 5% (weak distinctive component), 15% (intermediate strong) and 30% (strong). In Table 1, in which the amount of explained variance for each component is displayed, one can see that the ratio of explained variance for the other two (common) components is kept constant at $\frac{70\%}{30\%}$.

For each combination of the levels of the five manipulated factors, component matrices $\mathbf{A}^{(T)}$, $\mathbf{B}_1^{(T)}$, $\mathbf{C}^{(T)}$ and $\mathbf{B}_2^{(T)}$ were constructed by independently drawing entries from a standard normal

Table 1: Proportion of explained variance of the components underlying the common mode of the three-way and the two-way data block for each level of the amount of variance that the distinctive component explains.

	Common		Distinctive
Level	Component 1	Component 2	Component 3
Weak	.665	.285	.050
Intermediate	.595	.255	.150
Strong	.490	.210	.300

distribution; in a second step, these matrices were centered (i.e., mean of zero) and columnwise orthonormalized (i.e., orthogonal columns of length one). To ensure that the components underlying the two-way data block had the required proportions of explained variance (see Table 1), the columns of $\mathbf{B}_2^{(T)}$ were appropriately scaled. The entries for $\underline{\mathbf{G}}^{(T)}$ were independently drawn from a standard normal distribution and, next, rescaled such that the sums of squared entries of the rows of $\underline{\mathbf{G}}^{(T)}$ were in accordance with the required proportions of explained variance of the components for the common mode (see Table 1)[8]. To add a distinctive component to each data block, we augmented $\mathbf{A}^{(T)}$ with a different column (yielding $\mathbf{A}_{3D}^{(T)}$ and $\mathbf{A}_{2D}^{(T)}$, which only differ in the last column) such that each augmented $\mathbf{A}^{(T)}$ had been centered and columnwise orthonormalized, the correlation between the distinctive component in $\mathbf{A}_{3D}^{(T)}$ and $\mathbf{A}_{2D}^{(T)}$ was zero and the amounts of explained variance of the components was as required (see Table 1). Next, the true block couple ($\underline{\mathbf{T}}^1$, $\mathbf{T}^2$) was computed by combining the true parameters in $\mathbf{A}^{(T)}$, $\mathbf{B}_1^{(T)}$, $\mathbf{C}^{(T)}$, $\mathbf{B}_2^{(T)}$ and $\underline{\mathbf{G}}^{(T)}$ by the $Tucker3$ and PCA decomposition rules in (1) and (2). Further, for each true block couple ($\underline{\mathbf{T}}^1$, $\mathbf{T}^2$), a data block couple ($\underline{\mathbf{D}}^1$, $\mathbf{D}^2$) was constructed by adding error matrices $\underline{\mathbf{E}}^1$ $(I \times J \times K)$ and $\mathbf{E}^2$ $(I \times L)$ to $\underline{\mathbf{T}}^1$ and $\mathbf{T}^2$, respectively; $\underline{\mathbf{E}}^1$ and $\mathbf{E}^2$ were randomly generated, sampling from a standard normal distribution for each error entry, and rescaled in order to obtain the required amount of noise ε which was held constant for both data blocks (for more information regarding this rescaling procedure, see Wilderjans et al., 2009b). The whole data generation procedure was repeated 20 times to obtain 20 replications per cell of the design.

Fully crossing all design factors resulted in 2 (size three-way data) $\times$ 2 (ratio) $\times$ 2 (rank) $\times$ 2 (amount of noise) $\times$ 3 (strength of distinctive component) $\times$ 20 (replication) $=$ 960 different true block couples ($\underline{\mathbf{T}}^1$, $\mathbf{T}^2$) and data block couples ($\underline{\mathbf{D}}^1$, $\mathbf{D}^2$). After pre-processing the data (see Section 2.1), the $T3$-PCA algorithm was applied to each data block couple ($\underline{\mathbf{D}}^1$, $\mathbf{D}^2$), using the true rank $(P \times Q \times R)$ and repeating the analysis with ten different α-values: .050, .100, .300, .500, .600, .700, .800, .900, .950 and .999. In order to compare the integrated with the segmented strategy, each (pre-processed) generated coupled data set was further subjected to the following segmented procedure: first, a $Tucker3$ analysis was applied to $\underline{\mathbf{D}}^1$ and, next, a PCA was performed to $\mathbf{D}^2$ under the constraint of $\mathbf{A}$ being the same as in the $Tucker3$ analysis. To increase the probability of

[8]For $Tucker3$, when the component matrices $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$ are orthonormal, the squared entries of the core array $\underline{\mathbf{G}}$ are proportional to the amount of variation explained by each combination of components. As such, the sums of squared entries of the rows of $\underline{\mathbf{G}}$ are related to the amount of explained variance of each component in $\mathbf{A}$ (Smilde et al., 2004; Kiers and Van Mechelen, 2001).

the $T3$-PCA and the $Tucker3$ algorithm not retaining a local optimal solution, a multistart procedure with one rational and 50 random starts was adopted. The simulation study was performed in R (version 3.0.2) on a cluster of computers from the Flemish Supercomputer Center (VSC).

4.3 Results

4.3.1 Optimization of the loss function: sensitivity to local minima

The goal of this subsection is to investigate whether the $T3$-PCA algorithm is capable of identifying the global optimum of the loss function in (3). From the moment, however, that noise has been added to the data, the optimal solution is always unknown (see Wilderjans et al., 2008). A way out consists of defining a proxy for the global optimum, which can be conceived as our best guess of the global optimal solution, and evaluating whether or not the $T3$-PCA algorithm yields a solution that has a loss function value that equals or is lower than the loss function value for the proxy. We define the proxy as the solution with the lowest loss function value encountered among the following two $T3$-PCA solutions: (1) the true solution $\mathbf{A}^{(T)}$, $\mathbf{B}_1^{(T)}$, $\mathbf{C}^{(T)}$, $\underline{\mathbf{G}}^{(T)}$ and $\mathbf{B}_2^{(T)}$ underlying the data and (2) the solution obtained by using the segmented procedure. We consider a $T3$-PCA solution as suboptimal when its loss value exceeds the loss value of the proxy. This appeared to be the case for only 27 out of the $9,600$ analyses ($.28\%$), with all these cases belonging to the analyses with α being equal to .999.

4.3.2 Recovery of the model parameters

This subsection has a two-fold aim: (1) studying the recovery performance of the $T3$-PCA algorithm and (2) investigating whether and to which extent the integrated $T3$-PCA strategy outperforms the segmented procedure. Regarding both aims, we will especially focus on the recovery of the common component matrix **A**. To this end, recovery performance was measured by computing a goodness-of-recovery (GOR) statistic for each component matrix. In particular, for the common component matrix **A**, GOR was defined as follows:

$$GOR_{\mathbf{A}} = \frac{\sum_{p=1}^{P} |\phi(a_p^{(T)}, a_p^{(M)})|}{P} \tag{5}$$

with ϕ being the Tucker phi-coefficient (Tucker, 1951), $a_p^{(T)}$ and $a_p^{(M)}$ the p^{th} component of the true loading matrix $\mathbf{A}^{(T)}$ and the loading matrix $\mathbf{A}^{(M)}$ resulting from the $T3$-PCA analysis, respectively, and P the number of components. To account for the rotational freedom of the components (see Section 3.4), before computing $GOR_{\mathbf{A}}$, $\mathbf{A}^M$ was orthogonally rotated towards $\mathbf{A}^T$. A similar GOR statistic was computed for $\mathbf{B}_1$, **C** and $\mathbf{B}_2$. The GOR-statistics yield values between 0 and 1, with 1 indicating perfect recovery and 0 that recovery is at chance level.

The $T3$-PCA algorithm In Table 2, the mean GOR-value for **A**, $\mathbf{B}_1$, **C** and $\mathbf{B}_2$ is displayed, computed across all analyses and for each α-level separately. From this table, we can see that, on average, $\mathbf{B}_1$ and **C** are recovered almost perfectly and that **A** and $\mathbf{B}_2$ are disclosed to a large

extent (i.e., a ϕ-value larger than .85 indicates good recovery and a ϕ-value exceeding .95 implies excellent recovery, see Lorenzo-Seva and ten Berge, 2006). When inspecting the results for the different values of α, it appears that the recovery of all parameters increases when α increases from .050 till .500. For larger α-values, however, the recovery of **A** and $\mathbf{B}_2$ decreases, whereas recovery stays excellent for $\mathbf{B}_1$ and **C**.

Table 2: Mean recovery value for the component matrices (GOR), computed across all datasets and analyses and for each α-level separately.

α-level	$GOR_{\mathbf{A}}$	$GOR_{\mathbf{B}_1}$	$GOR_{\mathbf{C}}$	$GOR_{\mathbf{B}_2}$
.050	.85	.95	.96	.79
.100	.93	.91	.97	.80
.300	.96	.96	.99	.90
.500	.98	.96	.99	.93
.600	.96	.96	.99	.93
.700	.94	.96	.99	.92
.800	.92	.96	.99	.91
.900	.90	.96	.99	.90
.950	.88	.96	.99	.90
.999	.88	.96	.99	.83
Overall	.91	.96	.99	.89

To study how the recovery performance varies as a function of the five design factors and the α-value, we performed four repeated measures analysis of variances ($RMANOVA$), one for each GOR-measure, with the recovery measure as the dependent variable and the five factors (i.e., between factors) and the α-value (i.e., within factor) as independent variables. From these analyses, only considering sizeable effects (i.e., with $\eta^2_G \geq 0.35$, see Bakeman, 2005), it appears that most of the variation in $GOR_{\mathbf{A}}$ and $GOR_{\mathbf{B}_2}$ is explained by a large main effect of the strength of the (extra) distinctive component ($\eta^2_G = .73$ and $\eta^2_G = .65$ respectively): recovery performance decreases when the distinctive component becomes more important in terms of explained variance. This main effect, however, is qualified by a considerable interaction between the α-value and the strength of the distinctive component ($\eta^2_G = .54$ and $\eta^2_G = .71$ respectively). In particular, as one can see in Figure 2, when the distinctive component is not very strong (i.e., 5% and 15% of explained variance), recovery decreases when α increases. Moreover, the difference in recovery between datasets with a small versus intermediate strong distinctive component increases when α gets larger. However, when there is a strong distinctive component (i.e., explaining 30%), recovery is optimal for intermediate α-values and decreases when α goes to 0 (i.e., all weights on the two-way data block) and 1 (i.e., only the information in the three-way data block has an influence on the analysis), with the decrease being smaller for the latter than for the former. For $GOR_{\mathbf{B}_1}$ and $GOR_{\mathbf{C}}$, because of limited variability, there are no sizeable effects.

Because, as can be seen in Table 2, optimal recovery values are obtained when α equals .50 (i.e., both data blocks have the same amount of influence on the analysis), we zoomed in on the analyses with $\alpha = .50$. Performing an analysis of variance ($ANOVA$) with each recovery mea-

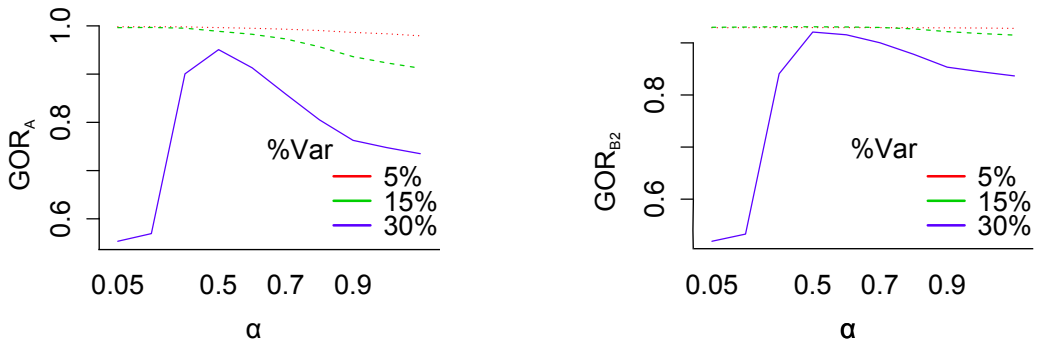


Figure 2: Mean $GOR_{\mathbf{A}}$ (left-hand panel) and $GOR_{\mathbf{B}_2}$ (right-hand panel) as a function of α and the strength of the distinctive component.

sure in turn as dependent variable and the five manipulated design factors as independent variables and only focusing on effects with an intraclass correlation $\hat{p}_I$ (Haggard, 1958; Kirk, 1982) larger than .05, the following important effects were revealed: first, the recovery of **A** decreases when the distinctive component becomes stronger ($\hat{p}_I = 0.39$) as the mean $GOR_{\mathbf{A}}$-value equals .996, .989 and .951 when the distinctive component is weak, intermediate strong and strong, respectively. This main effect, however, is qualified by a sizeable interaction between the rank of the model and the importance of the distinctive component ($\hat{p}_I = 0.09$). In particular, the decrease in $GOR_{\mathbf{A}}$ for increasing importance of the distinctive component is less pronounced when the rank is higher. Second, the recovery performance of $\mathbf{B}_1$ increases when the rank increases ($\hat{p}_I = .14$; mean $GOR_{\mathbf{B}_1}$ of .951 and .965 for the rank being $(2, 2, 2)$ and $(2, 3, 4)$, respectively) and when the amount of noise in the data increases ($\hat{p}_I = 0.21$; mean $GOR_{\mathbf{B}_1}$ of .950 and .966 for ε equal to .15 and .30, respectively).

Recovery difference between the integrated $T3$-PCA and the segmented strategy In this subsection, we will compare the integrated $T3$-PCA strategy to the segmented strategy (i.e., first fitting a $Tucker3$ model to the three-way data and next fitting a PCA to the two-way data under the constraint of equal **A**, see Section 4.2) in terms of uncovering the true component matrices. To this end, we computed for each recovery measure and for each analysis, the difference in recovery value between the integrated and the segmented strategy (e.g., for **A**, $GOR_{\mathbf{A}}^{diff} = GOR_{\mathbf{A}}^{T3-PCA} - GOR_{\mathbf{A}}^{segmented}$). The mean recovery difference value for all recovery measures is displayed in Table 3, computed across all analysis and for each level of α separately. From this table, it can be seen that the segmented strategy outperforms $T3$-PCA when the information in the two-way data block dominates the analysis (i.e., $\alpha \leq .10$). However, when the influence of both data blocks in the analysis is more balanced, the underlying parameters are better disclosed by $T3$-PCA than by

the segmented strategy. In particular, this is especially true for the recovery of the common mode **A** and to a lesser extent for $\mathbf{B}_2$, whereas no recovery differences are encountered for $\mathbf{B}_1$ and **C**. When the three-way data block dominates the analysis, both strategies perform equally well. The largest recovery differences are observed when both data blocks have an equal amount of influence on the analysis (i.e., $\alpha = .50$).

Table 3: Mean recovery difference between the integrated $T3\text{-}PCA$ and the segmented strategy for the component matrices (GOR^{diff}), computed across all datasets and analysis and for each level of α separately.

α-level	$GOR_{\mathbf{A}}^{diff}$	$GOR_{\mathbf{B}_1}^{diff}$	$GOR_{\mathbf{C}}^{diff}$	$GOR_{\mathbf{B}_2}^{diff}$
.050	-.03	-.08	-.03	-.10
.100	-.02	-.01	-.03	-.10
.300	.09	.00	-.03	.01
.500	.10	.00	.00	.03
.600	.09	.00	.00	.03
.700	.07	.00	.00	.03
.800	.04	.00	.00	.02
.900	.01	.00	.00	.01
.950	.01	.00	.00	.00
.999	.00	.00	.00	.00
Overall	.04	.00	-.01	-.01

To determine the influence of α and the design factors on the recovery difference between the integrated and the segmented strategy, we performed five $RMANOVA$s with the recovery difference as the dependent variable and the five factors and the α-value as independent variables. When only focusing on sizeable effects (i.e., $\eta_G^2 \geq 0.35$, Bakeman, 2005), it appears that $T3\text{-}PCA$ outperforms the segmented strategy in terms of disclosing the common components in **A** and $\mathbf{B}_2$ to a larger extent when the information in both data blocks is (more or less) equally weighted (η_G^2 equals .33 and .48 for **A** and $\mathbf{B}_2$). As can be seen in Table 3, the largest $GOR_{\mathbf{A}}^{diff}$ and $GOR_{\mathbf{B}_2}^{diff}$ is encountered for intermediate values of α (i.e., around .50). Moreover, the recovery difference for both component matrices decreases when one data block starts to dominate the analysis, with this effect being stronger for the two-way data matrix (i.e., even negative mean GOR^{diff} values when $\alpha \leq .10$) than for the three-way data array. This main effect, however, is qualified by an interaction between the strength of the distinctive component and α (η_G^2 is .48 and .66 for **A** and $\mathbf{B}_2$). From Figure 3, one can see that $T3\text{-}PCA$ always uncovers the common mode better than the segmented strategy when $\alpha \geq .50$, with the difference between both strategy being larger when the distinctive component is more influential. For low values of α, however, $T3\text{-}PCA$ only outperforms the segmented strategy when the distinctive component is not too strong, whereas the opposite is true when the distinctive component becomes dominant.

Because the largest recovery differences are encountered when both data blocks have an equal amount of influence on the analysis, we will further focus on the analyses with $\alpha = .50$. To study the effect of the manipulated factors on the recovery difference, four $ANOVA$s were performed

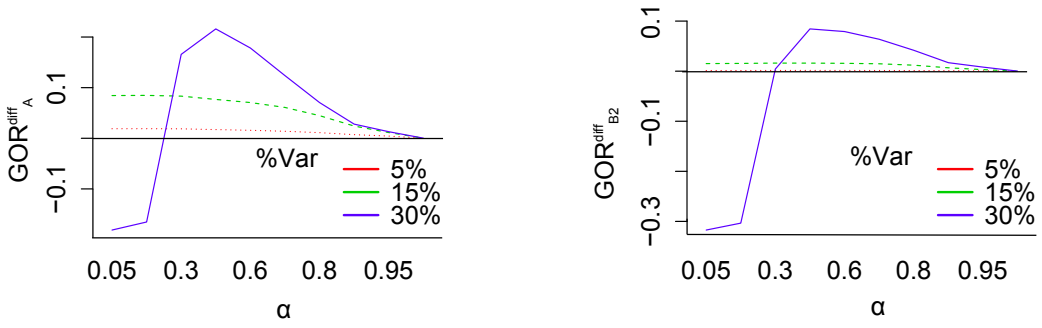


Figure 3: Mean $GOR_{\mathbf{A}}^{diff}$ (left-hand panel) and $GOR_{\mathbf{B}_2}^{diff}$ (right-hand panel) as a function of α and the strength of the distinctive component.

with the recovery difference as dependent variable and the design factors as independent variables. Only considering sizeable effects (i.e., $\hat{p}_I > .10$, see Haggard, 1958; Kirk, 1982), it appears that the size of the recovery differences for **A** and $\mathbf{B}_2$ mainly depend on the strength of the distinctive component and the rank of the model, whereas no important effects are observed for $\mathbf{B}_1$ and **C**. In Table 4, the mean $GOR_{\mathbf{A}}^{diff}$ and $GOR_{\mathbf{B}_2}^{diff}$ is displayed as a function of the rank of the model and the strength of the distinctive component. From this table, it appears that $T3\text{-}PCA$ outperforms the segmented strategy to a larger extent when the distinctive component becomes more influential ($\hat{p}_I$ is $.42$ and $.24$ for **A** and $\mathbf{B}_2$). Moreover, there is a considerable interaction between the rank of the model and the strength of the distinctive component ($\hat{p}_I$ equals $.15$ and $.11$ for **A** and $\mathbf{B}_2$): As can be seen in Table 4, when the rank increases and the distinctive component is weak or intermediate strong, the recovery difference becomes smaller; however, when the rank increases and there is a strong distinctive component, the recovery difference increases in favour of the integrated strategy.

Table 4: Mean $GOR_{\mathbf{A}}^{diff}$ and $GOR_{\mathbf{B}_2}^{diff}$ as a function of the rank of the model and the strength of the distinctive component, only considering the analyses with $\alpha = .50$.

	$GOR_{\mathbf{A}}^{diff}$			$GOR_{\mathbf{B}_2}^{diff}$		
Rank of the model	5%	15%	30%	5%	15%	30%
$(2, 2, 2)$	.029	.107	.174	.003	.028	.061
$(2, 3, 4)$	.005	.046	.259	.000	.004	.108
Overall	.017	.077	.216	.001	.016	.084

4.3.3 Discussion of the results

The results of the simulation study show that the $T3\text{-}PCA$ algorithm succeeds well in optimizing the loss function in (3) when taking the best start encountered among one rational and 50 random

starts. Focusing on recovery, it appears that the common components are disclosed to a (very) large extent by the $T3$-PCA algorithm. For the non-common modes, it is observed that the modes of the $Tucker3$ model are better recovered than the one of the PCA model. The best recovery results are obtained when both data blocks are given an equal amount of influence in the analysis (i.e., $\alpha = .50$) and this especially when there is a strong distinctive component. Moreover, in the latter case, recovery decreases when one data block starts to dominate the analysis, with this effect being more pronounced for the PCA block than for the $Tucker3$ block (see Figure 2). When comparing the integrated $T3$-PCA strategy to the segmented one, it appears that, in general, the integrated strategy better recovers the common mode and (to a smaller extent) the non-common PCA mode than the segmented strategies, whereas no differences between both strategies are encountered for the non-common $Tucker3$ modes. Moreover, the largest recovery differences are encountered when $\alpha = .50$. When there is a strong distinctive component, the recovery difference decrease when α goes to zero or one (i.e., domination of the PCA or $Tucker3$ part, respectively). When the PCA part starts to largely dominate the analysis, the segmented strategy even outperforms the integrated one (see Figure 3). It can be concluded that when the optimal rank of the $T3$-PCA model is known, solutions of a (very) good quality are delivered by the $T3$-PCA algorithm when using one rational and 50 random starts and an intermediate value of α. It may be advised to only use extreme α-values (i.e., around zero or one) when the researcher has good reasons to do so (e.g., being mainly interested in the structure underlying one data block) as recovery starts to decline in this case.

5 Concluding remarks

In this article, we proposed $T3$-PCA, a global model for an integrated analysis of a coupled data set consisting of a real-valued three-way three-mode data array that shares a single mode with a real-valued two-way two-mode data matrix. In $T3$-PCA, the three-way data array is modelled with $Tucker3$ and the two-way data matrix with PCA. The common mode is represented by a single set of common components that are estimated by using the information present in both data blocks (i.e., data fusion). The novel model is less restrictive than $LMPCA$ (Wilderjans et al., 2009b,a), which decomposes the three-way data according to $Parafac$, because the number of components is not restricted to be equal across the three modes and more complicated interactions between components of different modes are allowed. In an extensive simulation study, it has been demonstrated that the $T3$-PCA performs well in terms of optimizing the loss function and disclosing the structure underlying the coupled data set. Regarding both performance aspects, the integrated $T3$-PCA strategy, moreover, appears to clearly outperform a segmented strategy in which common components are determined by using the data available in a single data block only. In particular, by using information from both data blocks, the integrated strategy better recovers the common components and ignores components that are specific for one data set (i.e., distinctive components) than a segmented strategy, and, as a consequence, also better recovers the structure underlying the non-common modes of the coupled data set. These effects are optimal when both data blocks have the same amount of influence on the analysis (i.e., $\alpha = .50$).

In the remainder, generalizations of the proposed integrated modelling strategy are proposed,

making a distinction between modifications at the level of the model and at the level of the data analysis.

At the level of the model, first an extension of $T3$-PCA could be considered that can be used for the same type of coupled data. In $T3$-PCA it is assumed that all components underlying the common mode explain a considerable amount of variance in each data block. This, however, is a strict requirement as a considerable part of the information in each data block may pertain to aspects of the underlying processes that are specific for the data block in question. Although $T3$-PCA (by coincidence) may reveal "common" components that only explain a considerable amount of variance in a single data block, a better option, in order to capture a large amount of variance in each data block, is to allow some components to be distinctive for a data block (Schouteden et al., 2013; Van Deun et al., 2013; Schouteden et al., 2014; **?**). Therefore, it would be interesting to develop a model in which the common mode is represented by a combination of common and distinctive components. A good starting point for implementing this in the context of $T3$-PCA can be found in the work of Schouteden et al. (2013) and Van Deun et al. (2013).

The second model extension pertains to a different type of coupled data in which, besides additional information for the units of a single mode, also extra information for the other modes of the three-way data is available. For example, a researcher may collect data on behavioural signatures and at the same time information regarding characteristics of the persons (e.g., traits), responses (e.g., the extent to which an emotion is intra- or interpersonal) and situations (e.g., the extent to which a situation is experienced as threatening and/or whether or not the situation involves other people) under study. To disclose the processes underlying such coupled data, a generalization of $T3$-PCA may be developed in which the three-way data is modelled with $Tucker3$ and each of the two-way matrices (i.e., one for the persons, responses and situations) with PCA. Note that in this model, each mode of the three-way data is a common mode. An ALS algorithm for this novel global model could be constructed in which the common modes and the non-common PCA modes are updated alternatingly. To estimate each common mode, the information in the three-way array is integrated with the information of the associated two-way matrix.

At the level of the data analysis, two possible improvements could be considered. A first improvement pertains to the alternating least squares (ALS) algorithm to estimate the $T3$-PCA model parameters. In the context of $Parafac$, **?** show that an ALS algorithm does not recover the underlying structure well when a too complex model (i.e., with too many components, also denoted by overfactoring) is selected. To overcome this problem, **?** propose $CPOPT$, a gradient-based optimization approach in which all parameters are updated at once instead of per parameter set as in ALS, and show that $CPOPT$ is more robust again overfactoring than ALS. As selecting the model with an optimal complexity is a non-trivial task for $T3$-PCA (i.e., the optimal number of components should be determined for each mode separately), having an algorithm that is robust against overfactoring may ease this task a lot. Therefore, it may be interesting to implement an all-at-once optimization approach for $T3$-PCA. To this end, one may extend the $CMTF-OPT$ approach of **?**, which is an all-at-once approach for $LMPCA$, to $T3$-PCA.

Second, in $T3$-PCA it is assumed that the weight α that is used in loss function (3) to weight both data blocks is known beforehand. In many cases, however, it is hard to make an adequate choice for the value of α. One of the (main) factors determining which α-values are optimal

pertains to the amount of noise (e.g., measurement error[9]) that is present in each data block. Simulation studies in the context of coupled two-way data blocks, with each block being modelled with PCA, demonstrate that when data blocks have different amounts of noise (i.e., noise heterogeneity), the underlying structure is best recovered when noisy data blocks are downweighted in favour of less noisy ones (Wilderjans et al., 2011; van den Berg et al., 2009). For the type of coupled data considered in this paper, differences in amount of noise can be expected when the information in both data blocks is measured with a different level of reliability (e.g., one block pertaining to physiological measures and the other one to psychological questionnaires, with the former being measured more reliably than the latter). As such, the $T3$-PCA strategy may be extended by estimating the amount of noise in each data block during the analysis and choosing α in such a way that noisy data blocks are downweighted. To this end, an alternating algorithm may be developed in which the underlying structure (i.e., component matrices and core) and the noise are estimated in turn. Assuming residuals being independent and normally distributed, an optimal estimate, from a maximum likelihood point of view, for the amount of noise in each data block can be obtained by computing for each block the average of the squared residuals (see Wilderjans et al., 2011; **?**). It should be noted that differences in the amount of noise may not be limited to the block-level but may also be present at the level of the variables within each block. As such, a similar extension of $T3$-PCA to the case of noise heterogeneity among the variables within a block may also be considered (for a similar approach, see Wilderjans et al., 2012b).

References

Acar, E., Plopper, G., Yener, B., 2012. Coupled analysis of in vitro and histology tissue samples to quantify structure function relationship. PLoS One.

Aerts, S., Lambrechts, D., Maity, S., Van Loo, P., De Smet, F., Tranchevent, L., De Moor, B., Marynen, P., Hassan, B., Carmeliet, P., Moreau, Y., 2006. Gene prioritization through genomic data fusion. Nature Biotechnology 24, 537–544.

Andersson, C. A., Bro, R., 1998. Improving the speed of multi-way algorithms: Part i. tucker3. Chemometrics and Intelligent Laboratory Systems 42, 93–103.

Bakeman, R., 2005. Recommended effect size statistics for repeated measures designs. Behavior Research Methods 37, 379–384.

Boque, R., Smilde, A. K., 1999. Monitoring and diagnosing batch processes with multiway covariates regression models. AIChE Journal 45, 1504–1520.

Bro, R., Andersson, C. A., 1998. Improving the speed of multiway algorithms: Part ii. compression. Chemometrics and Intelligent Laboratory Systems 42, 105–113.

[9]Note that, besides measurement error, noise may also be present because of weak components that only explain a small amount of variance and that are not relevant for the processes under study (**?**).

Bro, R., Smilde, A. K., 2003. Centering and scaling in component analysis. Journal of Chemometrics 17 (1), 16–33.

Carroll, J. D., Chang, J. J., 1970. Analysis of individual differences in multidimensional scaling via an n-way generalization of "eckart-young" decomposition. Psychometrika 35, 283–319.

Ceulemans, E., Kiers, H. A. L., 2006. Selecting among three-mode principal component models of different types and complexities: A numerical convex hull based method. British Journal of Mathematical and Statistical Psychology 59, 133–150.

Ceulemans, E., Van Mechelen, I., Leenen, I., 2007. The local minima problem in hierarchical classes analysis: An evaluation of a simulated annealing algorithm and various multistart procedures. Psychometrika 72, 377–391.

Chiang, J., Wang, Z. J., McKeown, M. J., 2010. Multiblock pls model for group corticomuscular activity analysis in parkinson disease. In: Matthews, M. B. (Ed.), Conference Record of the Forty-Fourth Asilomar Conference on Signals, Systems and Computers (ASILOMAR), Pacific Grove, California, US, 7-10 November 2010. The Institute of Electrical and Electronics Engineers (IEEE), Inc., New York, US, pp. 1375–1379.

Coxe, K. L., 1986. Principal components regression analysis. In: Johnson, N. L., Kotz, S. (Eds.), Encyclopedia of statistical science (Volume 7). Wiley, New York, US, pp. 181–186.

Curran, P., Hussong, A., 2009. Integrative data analysis: the simultaneous analysis of multiple data sets. Psychological Methods 14, 81–100.

de Leeuw, J., 1994. Block-relaxation algorithms in statistics. In: Bock, H., Lenski, W., Richter, M. M. (Eds.), Information Systems and Data Analysis. Springer Verlag, Berlin, pp. 308–325.

Ermis, B., Acar, E., Cemgil, A., 2012. Link prediction via generalized coupled tensor factorisation. ECML/PKDD Workshop on Collective Learning and Inference on Structured Data.

Haggard, E. A., 1958. Intraclass correlation and the analysis of variance. Dryden, New York.

Harshman, R. A., 1970. Foundations of the PARAFAC procedure: Models and conditions for an explanatory multi-modal factor analysis. UCLA Working Papers in Phonetics 16, 1–84.

Hitchcock, F. L., 1927. The expression of a tensor or a polyadic as a sum of products. Journal of Mathematical Physics 6, 164–189.

Hotelling, H., 1933. Analysis of a complex of statistical variables into principal components. Journal of Educational Psychology 24, 417–441.

Jolliffe, I. T., 2002. Principal component analysis. Second ed. Springer Series in Statistics. Springer-Verlag, New York.

Kaiser, H. F., 1958. The varimax criterion for analytic rotation in factor analysis. Psychometrika 23, 187–200.

Kiers, H. A. L., 1992. Tuckals core rotations and constrained tuckals modelling. Statistica Applicata 4, 659–667.

Kiers, H. A. L., 1997. Three-mode orthomax rotation. Psychometrika 62, 579–598.

Kiers, H. A. L., 2000. Towards a standardized notation and terminology in multiway analysis. Journal of Chemometrics 14, 105–122.

Kiers, H. A. L., Van Mechelen, I., 2001. Three-way component analysis: Principles and illustrative application. Psychological Methods 6, 84–110.

Kirk, R. E., 1982. Experimental design: Procedures for the behavioral sciences, 2nd Edition. Brooks/Cole, Belmont, CA.

Kroonenberg, P. M., 1983. Three-mode Principal Component Analysis: Theory and applications. DSWO, Leiden.

Kroonenberg, P. M., 2008. Applied multiway data analysis. Wiley, Hoboken, NJ.

Kroonenberg, P. M., De Leeuw, J., 1980. Principal component analysis of three-mode data by means of alternating least squares algorithms. Psychometrika 45, 69–97.

Kroonenberg, P. M., ten Berge, J. M. F., 2011. The equivalence of tucker3 and parafac models with two components. Chemometrics and Intelligent Laboratory Systems 106, 21–26.

Lin, Y., Sun, J., Castro, P., Konuru, R., Sundaram, H., Kelliher, A., 2009. Metafac: community discovery via relational hypergraph factorization. KDD'09: Proc. 15th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, 527–536.

Lorenzo-Seva, U., ten Berge, J. M. F., 2006. Tucker's congruence coefficient as a meaningful index of factor similarity. Methodology 2 (2), 57–64.

Mitchell, T. M., Shinkareva, S. V., Carlson, A., Chang, K. M., Malave, V. L., Mason, R. A., Just, M. A., 2008. Predicting human brain activity associated with the meanings of nouns. Science 320 (5880), 1191–1195.

Palatucci, M., Pomerleau, D., Hinton, G., Mitchell, T. M., 2009. Zero-shot learning with semantic output codes. Advances in Neural Information Processing Systems 22, 1410–1418.

Pearson, K., 1901. On lines and planes of closest fit to systems of points in space. Philosophical Magazine, Series 6 2, 559–572.

Schouteden, M., Van Deun, K., Pattyn, S., Van Mechelen, I., 2013. Sca with rotation to distinguish common and distinctive information in linked data. Behavior Research Methods 45, 822–833.

Schouteden, M., Van Deun, K., Wilderjans, T. F., Van Mechelen, I., 2014. Performing disco-sca to search for distinctive and common information in linked data. Behavior Research Methods 46, 576–587.

Smilde, A., Bro, R., Geladi, P., 2004. Multi-way analysis with applications in the chemical sciences. Wiley, Chichester, UK.

Smilde, A., van der Werf, M., Bijlsma, S., van der Werff-van-der Vat, B., Jellema, R., 2005. Fusion of mass spectrometry-based metabolomics data. Analytical Chemistry 77, 6729–6736.

Smilde, A. K., Kiers, H. A. L., 1999. Multiway covariates regression models. Journal of Chemometrics 13, 31–48.

Smilde, A. K., Westerhuis, J. A., Boqué, R., 2000. Multiway multiblock component and covariates regression models. Journal of Chemometrics 14, 301–331.

Smilde, A. K., Westerhuis, J. A., de Jong, S., 2003. A framework for sequential multiblock component methods. Journal of Chemometrics 17, 323–337.

ten Berge, J. M. F., 1986. Rotation to perfect congruence and the cross-validation of component weights across populations. Multivariate Behavioral Research 21, 41–64.

ten Berge, J. M. F., 1993. Least squares optimization in multivariate analysis. DSWO Press, Leiden.

Timmerman, M. E., Kiers, H. A. L., 2003. Four simultaneous component models for the analysis of multivariate time series from more than one subject to model intraindividual and interindividual differences. Psychometrika 68, 105–121.

Tucker, L. R., 1951. A method for synthesis of factor analysis studies. Personnel research section rapport # 984, Washington, DC: Department of the Army.

Tucker, L. R., 1964. The extension of factor analysis to three-dimensional matrices. In: Frederiksen, N., Gulliksen, H. (Eds.), Contributions to mathematical psychology. Holt, Rinehart and Winston, New York, USA, pp. 109–127.

Tucker, L. R., 1966. Some mathematical notes on three-mode factor analysis. Psychometrika 31, 279–311.

van den Berg, R. A., Van Mechelen, I., Wilderjans, T. F., Van Deun, K., Kiers, H. A. L., Smilde, A. K., 2009. Integrating functional genomics data using maximum likelihood based simultaneous component analysis. BMC Bioinformatics 10, 340.

Van Deun, K., Smilde, A. K., Thorrez, L., Kiers, H. A. L., Van Mechelen, I., 2013. Identifying common and distinctive processes underlying multiset data. Chemometrics and Intelligent Laboratory Systems 129, 40–51.

Van Mechelen, I., Schepers, J., 2007. A unifying model involving a categorical and/or dimensional reduction for multimode data. Computational Statistics and Data Analysis 52, 537–549.

Van Mechelen, I., Smilde, A., 2010. A generic linked-mode decomposition model for data fusion. Chemometrics and Intelligent Laboratory Systems 104, 83–94.

Vervloet, M., Van Deun, K., Van den Noortgate, W., Ceulemans, E., 2013. On the selection of the weighting parameter value in principal covariates regression. Chemometrics and Intelligent Laboratory Systems 123, 36–43.

Wansbeek, T., Verhees, J., 1989. Models for multidimensional matrices in econometrics and psychometrics. In: Coppi, R., Bolasco, S. (Eds.), Multiway data analysis. Elsevier, Amsterdam, pp. 543–552.

Wilderjans, T., Ceulemans, E., Meers, K., 2013. Chull: A generic convex-hull-based model selection method. Behavior Research Methods 45, 1–15.

Wilderjans, T. F., Ceulemans, E., 2013. Clusterwise parafac to identify heterogeneity in three-way data. Chemometrics and Intelligent Laboratory Systems 129, 87–97.

Wilderjans, T. F., Ceulemans, E., Kiers, H. A. L., Meers, K., 2009a. The lmpca program: A graphical user interface for fitting the linked-mode parafac-pca model to coupled real-valued data. Behavior Research Methods 41, 1073–1082.

Wilderjans, T. F., Ceulemans, E., Kuppens, P., 2012a. Clusterwise hiclas: A generic modeling strategy to trace similarities and differences in multi-block binary data. Behavior Research Methods 44, 532–545.

Wilderjans, T. F., Ceulemans, E., Van Mechelen, I., 2008. The chic model: A global model for coupled binary data. Psychometrika 73, 729–751.

Wilderjans, T. F., Ceulemans, E., Van Mechelen, I., 2009b. Simultaneous analysis of coupled data blocks differing in size: A comparison of two weighting schemes. Computational Statistics and Data Analysis 53, 1086–1098.

Wilderjans, T. F., Ceulemans, E., Van Mechelen, I., 2012b. The simclas model: Simultaneous analysis of coupled binary data matrices with **noise** heterogeneity between and within data blocks. Psychometrika 77, 724–740.

Wilderjans, T. F., Ceulemans, E., Van Mechelen, I., van den Berg, R. A., 2011. Simultaneous analysis of coupled data matrices subject to different amounts of noise. British Journal of Mathematical and Statistical Psychology 64, 277–290.

Zheng, V., Cao, B., Zheng, Y., Xie, X. Yang, Q., 2010. Collaborative filtering meets mobile recommendation: A user-centered approach. AAAI'10: Proc. 24th Conf. on Artificial Intelligence, 236–241.

BIBLIOGRAFIA

(2011). R: A language and environment for statistical computing. r foundation for statistical computing.

Abdallah, E. E., Hamza, A. B., and Bhattacharya, P. (2007). Mpeg video watermarking using tensor singular value decomposition. In *Image Analysis and Recognition, Lecture Notes in Comput. Sci. 4633, Springer*.

Acar, E., Bingol, C. A., Bingol, H., Bro, R., and Yener, B. (2007). Multiway analysis of epilepsy tensors. *Bioinformatics*, 23:10–18.

Acar, E., Camtepe, S. A., Krishnamoorthy, M. S., and Yener, B. (2005). Modeling and multiway analysis of chatroom tensors. In *Proceedings of the IEEE International Conference on Intelligence and Security Informatics, Lecture Notes in Comput. Sci. 3495, Springer*.

Acar, E., Camtepe, S. A., and Yener, B. (2006). Collective sampling and analysis of high order tensors for chatroom communications. In *Proceedings of the IEEE International Conference on Intelligence and Security Informatics, Lecture Notes in Comput. Sci. 3975, Springer*.

Acar, E., Plopper, G., and Yener, B. (2012). Coupled analysis of in vitro and histology tissue samples to quantify structure-function relationship. *PLoS One*.

Acar, E., Rasmussen, M., Savorani, F., Naes, T., and Bro, R. (2013). Understanding data fusion within the framework of coupled matrix and tensor factorizations. *Chemometrics and Intelligent Laboratory Systems*, 129:53–63.

Andersen, A. H. and Rayens, W. S. (2004). Structure-seeking multilinear methods for the analysis of fmri data. *NeuroImage*, 22:728–739.

Andersen, C. M. and Bro, R. (2003). Practical aspects of parafac modeling of fluorescence excitation-emission data. *J. Chemometrics*, 17:200–215.

Andersson, C. A. and Bro, R. (1998). Improving the speed of multi-way algorithms: Part i. tucker3. *Chemometrics and Intelligent Laboratory Systems*, 42:93–103.

Appellof, C. J. and Davidson, E. R. (1981). Strategies for analyzing data from video fluorometric monitoring of liquid chromatographic effluents. *Anal. Chem.*, 53:2053–2056.

Bader, B. W., Berry, M. W., and Browne, M. (2007). Discussion tracking in enron email using parafac. In *Survey of Text Mining:C lustering, Classification, and Retrieval, 2nd ed., M. W. Berry and M. Castellanos, eds., Springer.*

Bakeman, R. (2005). Recommended effect size statistics for repeated measures designs. *Behavior Research Methods*, 37:379–384.

Barbut, M. and Monjardet, B. (1970). *Ordre et classification: Algebre et combinatorie.*

Bauckhage, C. (2007). Robust tensor classifiers for color object recognition. In *Image Analysis and Recognition, Lecture Notes in Comput. Sci. 4633, Springer.*

Bradu, D. and Gabriel, K. R. (1978). The biplot as a diagnostic tool for models of two-way tables. *Technometrics*, 20:47–68.

Bro, R. (1988). *Mutiway analysis in the food industry. Models, algorithms and applications.* Tesis doctoral, University of Amsterdam.

Bro, R. and Andersson, C. A. (1998). Improving the speed of multiway algorithms: Part ii. compression. *Chemometrics and Intelligent Laboratory Systems*, 42:105–113.

Bro, R. and Kiers, H. A. L. (2003). A new efficient method for determining the number of components in parafac models. *Journal of Chemometrics*, 17:274–286.

Carlier, A. and Kroonenberg, P. M. (1996). Decompositions and biplots in three-way corresponde anaylisis. *Psychometrika*, 61:355–373.

Carroll, J. D. and Chang, J. (1970). Analysis of individual differences in multidimensional scaling via an n-way generalization of eckart-young descomposition. *Psychometrika*, 35:283–319.

Carroll, J. D., Pruzansky, S., and Kruskal, J. B. (1980). Candelinc: A general approach to multidimensional analysis of many-way arrays with linear constraints on parameters. *Psychometrika*, 45:3–24.

Cattell, R. B. (1966). *The meaning and strategic use of factor analysis.* In R.B. Cattell (Ed.), Handbook of multivariate experimental psychology (pp. 174-243). Chicago, If.: Rand McNally.

Ceulemans, E. and Kiers, H. A. L. (2006). Selecting among three mode principal component models of different types and complexities: A numerical convex hull based method. *British Journal of Mathematical and Statistical Psychology*, 59:133–150.

Ceulemans, E. and Kiers, H. A. L. (2009). Discriminating between strong and weak structures in three-mode principal component analysis. *British Journal of Mathematical and Statistical Psychology*, 62:601–620.

Ceulemans, E. and Van Mechelen, I. (2004). Tucker2 hierarchical classes analysis. *Psychometrika*, 69:375–399.

Ceulemans, E. and Van Mechelen, I. (2005). Hierarchical classes models for three-way three-mode binary data: Interrelations and model selection. *Psychometrika*, 70:461–80.

Ceulemans, E., Van Mechelen, I., and Leenen, I. (2003). Tucker3 hierarchical classes analysis. *Psychometrika*, 68:413–433.

Ceulemans, E., Van Mechelen, I., and Leenen, I. (2007). The local minima problem in hierarchical classes analysis: An evaluation of a simulated annealing algorithm and various multistart procedures. *Psychometrika*, 72:377–391.

Chaturvedi, A. and Carroll, J. D. (1994). An alternating combinatorial optimization approach to fitting the indclus and generalized indclus models. *Journal of Classification*, 11:155–170.

Choulakian, V. (1966). Generalized bilinear models. *Psychometrika*, 61(2):271–283.

Cooper, M. and DeLacy, I. H. (1994). Relationships among analytic methods used to study genotypic variation and genotype-byenvironment interaction in plant breeding multienvironment experiments. *Theor. Appl. Genet.*, 88:561–572.

Cornelius, P. L., Crossa, J., and Seyedsadr, M. S. (1996). Statistical tests and estimators for multiplicative models for genotype-by-environment interaction. *In Kang, M.S., Gauch, Jr. H.G. (eds.)* Genotype-by Environment Interaction. *CRC Press, Boca Raton, FL.*

Cornelius, P. L., Sanford, D. A. V., and Seyedsadr, M. (1993). Clustering cultivars into groups without rank-change interactions. *Crop Science*, 33:1193–1200.

Cornelius, P. L., Seyedsadr, M., and Crossa, J. (1992). Using the shifted multiplicative model to search for separability in crop cultivar trials. *Theoretical and Applied Genetics*, 84:161–172.

Cornelius, P. L. and Seyedsard, M. (1997). Estimation of general linear-bilinear models for two-way tables. journal of statistical computation and simulation. *Journal of Statistical Computation and Simulation*, 58:287–322.

Crossa, J. and Cornelius., P. L. (1997). Site regression and shifted multiplicative model clustering of cultivar trials sites under heterogeneity of error variances. *Crop Sci.*, 37:406–415.

Crossa, J., Cornelius, P. L., Sayre, K., and Ortiz-Monasterio, I. J. (1995). A shifted multiplicative model fusion method for grouping environments without cultivar rank change. *Crop Science*, 35:54–62.

Crossa, J., Cornelius, P. L., Seyedsadr, M., and Byrne, P. (1993). A shifted multiplicative model cluster analysis for grouping environments without genotypic rank-change. *Theoretical and Applied Genetics*, 85:577–586.

De Boeck, P. and Rosenberg, S. (1988). Hierarchical classes: Model anda data analysis. *Psychometrika*, 53:361–381.

Denis, J. B. (1991). Ajustements de modelles lineaires et bilineaires sous constraintes lineaires avec donnes manquantes. *Statistique Applique*, XXXIX(2):5–24.

Doostan, A., Iaccarino, G., and Etemadi, N. (2007). A least-squares approximation of high-dimensional uncertain systems. In *Annual Research Briefs, Center for Turbulence Research, Stanford University.*

Ebdon, J. S. and Gauch, H. G. (2002a). Additive main effect and multiplicative interaction analysis of national turfgrass performance trials i: Cultivar recommendations. *Crop Sci*, 42:497–506.

Ebdon, J. S. and Gauch, H. G. (2002b). Additive main effect and multiplicative interaction analysis of national turfgrass performance trials i: Interpretation of genotype x environment interaction. *Crop Sci*, 42:489–496.

Eberhart, S. A. and Russell, W. A. (1969). Yield stability for a 10-line diallel of single-cross and double-cross maize hybrids. *Crop Sci*, 9:357–361.

Escofier, B. and Pages, B. (1984). *L´analyse factorielle multiple: une m´ethode de comparaison de groupes de variables.* E. Diday, ed. Data Analysis and Informatics, III. Proccedings of the Third International Symposium of Data Analysis and Informatics.

Falguerolles, A. (1995). Generalized bilinear models and generalized biplots: Some examples. *Publications du Laboratoire de Statistique et Probabilites. Universite Paul Sabatier, Toulouse.*

Finlay, K. W. and Wilkinson, G. N. (1963). The analysis of adaptation in a plant-breeding programme. *Aust J Agric Res*, 14:742–754.

Fisher, R. A. and Mackenzie, W. A. (1923). The manurial response of different potato varieties. *Journal of Agricultural Science, Cambridge*, 23:311–320.

Furukawa, R., Kawasaki, H., Ikeuchi, K., and Sakauchi, M. (2002). Appearance based object modeling using texture database: Acquisition, compression and rendering. In *EGRW 02: Proceedings of the 13th Eurographics Workshop on Rendering, Aire-la-Ville, Switzerland, Eurographics Association.*

Gabriel, K. R. (1971). The biplot graphic display of matrices with application to principal component analysis. *Biometrika*, 58(3):453–467.

Gabriel, K. R. (1998). Generalised bilinear regression. *Biometrika*, 85:689–700.

Galindo, M. (1986). Una alternativa de representación simultanea: Hj-biplot. *Questíio*, 10(1):12–23.

Galindo, M. P. and Cuadras, C. (1986). Una extensión del método biplot y su relación con otras técnicas. *Publicaciones de Bioestadística y Biomatemática, 17. Barcelona: Universidad de Barcelona. PMCid:269031.*

Gauch, G. H. and Zobel, R. W. (1997). Interpreting mega-environments and targeting genotypes. *Crop Sci*, 37:311–326.

Gauch, H. G. (1988). Model selection and validation for yield trials with interaction. *Biometrics*, 44:705–715.

Gauch, H. G. (1992). *Statistical analysis of regional yield trials: AMMI analysis of factorial designs.* Elsevier, Amsterdam, The Netherlands.

Gauch, H. G. (2006). Statistical analysis of yield trials by ammi and gge. *Crop Sci.*, 46:1488–1500.

Gauch, H. G., Piepho, H. P., and Annicchiarico, P. (2008). Statistical analysis of yield trials by ammi and gge: Further considerations. *Crop Sci.*, 48:866–889.

Gauch, H. G. and Zobel, R. W. (1996). *AMMI analysis of yield trials.* CRC Press, Boca Raton, FL, US.

Golub, G. H. and Reinsch, C. H. (1970). Singular value decomposition and least squares solution. *Numer. Math.*, 14:403–420.

Gower, J. C. (1992). Generalized biplots. *Biometrika*, 79:475–493.

Gower, J. C. and Hand, D. J. (1996). *Biplots.* Chapman and Hall, London.

Gower, J. C. and Harding, S. A. (1988). Nonlinear biplots. *Biometrika*, 75:445–455.

Haggard, E. A. (1958). Intraclass correlation and the analysis of variance, dryden, new york.

Harshman, R. and Lundy, M. (1994). Parafac: parallel factor analysis. *Computational Statistics & Data Analysis*, 18:39–72.

Harshman, R. A. (1970). Foundations of the parafaf procedure: models and conditions for an explanatory multi-mode factor analysis. *UCLA Working Papers in Phonetics*, 16:1–84.

Harshman, R. A. (1972). Parafac2: Mathematical and technical notes. *UCLA Working Papers in Phonetics*, 22:30–47.

Harshman, R. A. (1978). Models for analysis of asymmetrical relationships among n objects or stimuli. In *First Joint Meeting of the Psychometric Society and the Society for Mathematical Psychology, McMaster University, Hamilton, Ontario.*

Harshman, R. A. (2005). The parafac model and its variants. Paper presented at Workshop on Tensor decompositions and applications, CIRM, Luminy, France.

Harshman, R. A. and Lundy, M. E. (1984). *The PARAFAC model for three-way factor analysis and multidimensional scaling.* New York: Praeger.

Harshman, R. A. and Lundy, M. E. (1996). Uniqueness proof for a family of models sharing features of tucker's three-mode factor analysis and parafac/candecomp. *Psychometrika*, 61:133–154.

Heijne, W., Lamers, R., Van Bladeren, P., Groten, J., Van Nesselrooij, J., and Van Ommen, B. (2005). Profiles of metabolites and gene expression in rats with chemically induced hepatic necrosis. *Toxicology Pathology*, 33:425–433.

Henrion, R. (1994). N-way principal component analysis theory, algorithms and applications. *Chemometrics and Intelligent Laboratory Systems*, 25:1–23.

Hernández, J. M., Polo, A., and Pozo, C. (1996). Inventario de estrés académico. *Servicio de Psicología Aplicada. U.A.M.*

Hitchcock, F. L. (1927). The expression of a tensor or a polyadic as a sum of products. *J. Math.Phys.*, 6:164–189.

Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24:417–441.

Householder, A. S. and Young, G. (1938). Matrix approximation and latent roots. *Am. Math. Monthly*, 45:165–171.

Jaccard, P. (1908). Nouvelles recherches sur la distribution florale. *Bulletin de la Societe Vaudoise Sciences Naturelles*, 44:223–270.

Kaiser, H. F. (1958). The varimax criterion for analytic rotation in factor analysis. *Psychometrika*, 23:187–200.

Kang, M. S. (1988). Using genotype-by-environment interaction for crop cultivar development. *Advances in Agronomy*, 62:199–252.

Kapteyn, A., Neudecker, H., and Wansbeek, T. (1986). An approach to n-mode components analysis. *Psychometrika*, 51:269–275.

Kempton, R. A. (1984). The use of biplots in interpreting variety by environment interactions. *The Journal of Agricultural Science*, 103:123–135.

Kiers, H. A. L. (1992). Tuckals core rotations and constrained tuckals modelling. *Statistica Applicata*, 4:659–667.

Kiers, H. A. L. (1997). Three-mode orthomax rotation. *Pychometrika*, 62:579–598.

Kiers, H. A. L. (1998). Three-way simplimax for oblique rotation of the three-mode factor analysis core to simple structure. *Computational Statistics and Data Analysis*, 28:307–324.

Kiers, H. A. L. (2000). Towards a standardized notation and terminology in multiway analysis. *Journal of Chemometrics*, 14:105–122.

Kiers, H. A. L. and Kinderen, A. (2003). A fast method for choosing the numbers of components in tucker3 analysis. *British Journal of Mathematical and Statistical Psychology*, 56:119–125.

Kiers, H. A. L. and Mechelen, I. V. (2001). Three-way component analysis: Principles and illustrative application. *Psych. Methods*, 6:84–110.

Kirk, R. E. (1982). Experimental design: Procedures for the behavioral sciences, 2nd edition brooks/cole, belmont, ca, 1982.

Kolda, T. G. (2006). Multilinear operators for higher-order decompositions. Technical report, Tech. Report SAND2006-2081, Sandia National Laboratories, Albuquerque, NM, Livermore, CA.

Kolda, T. G. and Bader, B. W. (2009). Tensor decompositions and applications. *SIAM Rev*, 51:455–500.

Krijnen, W. P. and Kroonenberg, P. M. (2000). Degeneracy and parafac. Technical report, Groningen, The Netherlands: Department of Psychology, University of Groningen.

Kroonenberg, P. M. (1983). *Three-Mode Principal Components Analysis. Theory and Applications.* DSWO-Press, Leiden.

Kroonenberg, P. M. (2008). *Applied Multiway Data Analysis.* John Wiley and Sons, New Jersey.

Kroonenberg, P. M. and de Leeuw, J. (1980). Principal component analysis of three-mode data by means of alternating least squares algorithms. *Psychometrika*, 45:69–97.

Kruskal, J. B. (1977). Three-way arrays: Rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics. *Linear Algebra Appl*, 18:95–138.

Kruskal, J. B. (1989). *Rank, decomposition, and uniqueness for 3-way and N-way arrays.* Multiway Data Analysis, R. Coppi and S. Bolasco, eds., North-Holland, Amsterdam.

Kruskal, J. B., Harshman, R. A., and Lundy, M. E. (1989). *How 3-MFA data can cause degenerate PARAFAC solutions, among other relationships.* Coppi & S. Bolasco (Eds.).

Lathauwer, L. D. and Castaing, J. (2007). Tensor-based techniques for the blind separation of ds-cdma signal. *Signal Process.*, 87:322–336.

Lathauwer, L. D., Moor, B. D., and Vandewalle, J. (2000). A multilinear singular value decomposition. *SIAM J. Matrix Anal. Appl.*, 21:1253–1278.

Lathauwer, L. D. and Vandewalle, J. (2004). Dimensionality reduction in higher-order signal processing and rank-(r1,r2, . . . , rn) reduction in multilinear algebra. *Linear Algebra Appl.*, 391:31–55.

Leenen, I. and Van Mechelen, I. (1998). *A branch-and-bound algorithm for Boolean regression.* Berlin, Germany: Springer-Verlag.

Leenen, I., Van Mechelen, I., and De Boeck, E. (2001). Models for ordinal hierarchical classes analysis. *Psychometrika*, 66:389–404.

Leenen, I., Van Mechelen, I., De Boeck, R., and Rosenberg, S. (1999). Indclas: A three-way hierarchical classes model. *Psychometrika*, 64:9–24.

L'Hermeir des plantes, H. (1976). *Structuration des Tableaux a Trois Indices de la Statistique: Theorie et Application d'une Méthode d'Analyse Conjointe.* PhD thesis, Université des Sciences et Techniques du Languedoc.

Lorenzo-Seva, U. and ten Berge, J. M. F. (2006). Tucker's congruence coefficient as a meaningful index of factor similarity. *Methodology*, 2 (2):57–64.

López Herrera, H., Pedrosa, I., Vicente Galindo, M. P., Suárez-Álvarez, J., Galindo Villardón, M. P., and García Cueto, E. (2014). Multivariate analysis of burnout syndrome in latin-american priests. *Psicothema*, 26(2):227–234.

Madeira, S. and Oliveira, A. (2004). Biclustering algorithms for biological data analysis: a survey. *IEEE Transactions on Computational Biology and Bioinformatics*, 1:24–25.

Martinez-Montes, E., Valdes-Sosa, P. A., Miwakeichi, F., Goldman, R. I., and Cohen, M. S. (2004). Concurrent eeg/fmri analysis by multiway partial least squares. *NeuroImage*, 22:1023–1034.

Maslach, C. and Jackson, S. (1981). Maslach burnout inventory. *Palo Alto, California: Consulting Psychologists Press.*

McDonald, R. P. (1980). A simple comprehensive model for the analysis of covariance structures: Some remarks on applications. *British Journal of Mathematical and Statistical Psycholoy*, 3:161.

Miwakeichi, F., Martinez-Montes, E., Valdes-Sosa, P. A., Nishiyama, N., Mizuhara, H., and Yamaguchi, Y. (2004). Decomposing eeg data into space-time-frequency components using parallel factor analysis. *NeuroImage*, 22:1035–1045.

Mocks, J. (1988). Topographic components model for event-related potentials and some biophysical considerations. *IEEE Transactions on Biomedical Engineering*, 35:482–484.

Muti, D. and Bourennane, S. (2005). Multidimensional filtering based on a tensor approach. *Signal Process*, 85:2338–2353.

Nagy, J. G. and Kilmer, M. E. (2006). Kronecker product approximation in three-dimensional imaging applications. *IEEE Trans. Image Process*, 15:604–613.

Nelder, J. A. and Wedderburn, R. W. M. (1972). Generalized linear models. *Journal of the Royal Statistical Society. Series A (General)*, 135(3):370–384.

Omberg, L., Golub, G. H., and Alter, O., editors (2007). *A tensor higher-order singular value decomposition for integrative analysis of DNA microarray data from different studies, Proceedings of the National Academy of Sciences of the United States of America*, volume 104.

Paatero, P. (2000). Construction and analysis of degenerate parafac models. *Journal of Chemometrics*, 14:285–299.

Pearson, K. (1901). On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, 2:559–572.

Rao, C. R. and Mitra, S. (1971). *Generalized inverse of matrices and its applications*. Wiley, New York.

Savas, B. and Elden, L. (2007). Handwritten digit classification using higher order singular value decomposition. *Pattern Recog.*, 40:993–1003.

Schott, J. R. (1997). *Matrix Analysis for Statistics*. John Wiley & Sons, Chichester.

Schouteden, M., Van Deun, K., Pattyn, K., and Van Mechelen, I. (2013). Sca with rotation to distinguish common and distinctive information in linked data. *Behavior Research Methods*, 45:822–833.

Schouteden, M., Van Deun, K., Wilderjans, T. F., and Van Mechelen, I. (2014). Performing disco-sca to search for distinctive and common information in linked data. *Behavior Research Methods*, 46:576–587.

Seyedsadr, M. and L., C. P. (1992). Shifted multiplicative model for nonadditive two-way tables. *Communications in Statistics B. Simulation and Computation*, 21:807–822.

Shashua, A. and Levin, A. (2001). Linear image coding for regression and classification using the tensor-rank principle. In *CVPR 2001: Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*.

Sidiropoulos, N. and Bro, R. (2000). On the uniqueness of multilinear decomposition of n-way arrays. *Journal of Chemometrics*, 14:229–239.

Sidiropoulos, N., Bro, R., and Giannakis, G. (2000a). Parallel factor analysis in sensor array processing. *IEEE Trans. Signal Process*, 48:2377–2388.

Sidiropoulos, N. and Budampati, R. (2002). Khatri-rao space-time codes. *IEEE Trans. Signal Process.*, 50:2396–2407.

Sidiropoulos, N., Giannakis, G., and Bro, R. (2000b). Blind parafac receivers for ds-cdma systems. *IEEE Trans. Signal Process*, 48:810–823.

Smilde, A., Westerhuis, J., and de Jong, S. (2003). A framework for sequential multiblock component methods. *Journal of Chemometrics*, 17:323–337.

Smilde, A. K., Bro, R., and Geladi, P. (2004). *Multi-way analysis Applications in the chemical sciences*. John Wiley & Sons Ltd, Chichester, England.

Smilde, A. K. and Kiers, H. A. L. (1999). Multiway covariates regression models. *Journal of Chemometrics*, 13:31–48.

Smilde, A. K., Westerhuis, J. A., and Boque, R. (2000). Multiway multiblock component and covariates regression models. *Journal of Chemometrics*, 14:301–331.

Stegeman, A. (2006). Degeneracy in candecompparafac explained for p x p x 2 arrays of rank p+1 or higher. *Psychometrika*, 71:483–501.

Sun, J., Papadimitriou, S., and Yu, P. S. (2006). Window-based tensor analysis on high-dimensional and multi-aspect streams. In *ICDM 2006: Proceedings of the 6th IEEE Conference on Data Mining, IEEE Computer Society Press*.

T. Wansbeek, T. and Verhees, J. (1989). *Multiway data analysis*. Models for multidimensional matrices in econometrics and psychometrics. In R. Coppi & S. Bolasco (Eds.).

Ten Berge, J. M. F. (1993). *Least squares optimization in multivariate analysis*. DSWO Press, Leiden.

Ten Berge, J. M. F., de Leeuw, J., and Kroonenberg, P. M. (1987). Some additional results on principal components analysis of three-mode data by means of alternating least squares. *Psychometrika*, 52:183–191.

Ten Berge, J. M. F. and Sidiriopolous, N. D. (2002). On uniqueness in candecomp/parafac. *Psychometrika*, 67:399–409.

Timmerman, M. E., Ceulemans, E., Kiers, H. A. L., and Vichi, M. (2010). Factorial and reduced k-means reconsidered. *Computational Statistics and Data Analysis*, 54:1858–1871.

Timmerman, M. E. and Kiers, H. A. L. (2000). Three-mode principal components analysis. choosing the numbers of components and sensitivity to local optima. *British Journal of Mathematical and Statistical Psychology*, 53:1–16.

Tucker, L. R. (1951). A method for synthesis of factor analysis studies. personnel research section rapport 984, washington, dc: Department of the army.

Tucker, L. R. (1966). Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31:279–311.

Tukey, J. W. (1949). One degree of freedom for non-additivity. *Biometrics*, 5:232–242.

Ukkelberg, A. and Borgen, O. (1993). Outlier detection by robust alternating regressions. *Analytica Chimica Acta*, 277:489–494.

van den Berg, R. A., Van Mechelen, I., Wilderjans, T. F., Van Deun, K., Kiers, H. A. L., and Smilde, A. K. (2009). Integrating functional genomics data using maximum likelihood based simultaneous component analysis. *BMC Bioinformatics*, 10:340.

Van Deun, K., Smilde, A., Van der Werf, M., Kiers, H., and Van Mechelen, I. (2009). A structured overview of simultaneous component based data integration. *BMC Bioinformatics*, 10:246.

Van Deun, K., Smilde, A. K., Thorrez, L., Kiers, H. A. . L., and Van Mechelen, I. (2013). Identifying common and distinctive processes underlying multiset data. *Chemometrics and Intelligent Laboratory Systems*, 129:40–51.

Van Eeuwijk, F. (1995). Multiplicative interaction in generalized linear models. *Biometrics*, 51:1017–1032.

Van Mechelen, I. (1988). Prediction of a dichotomous criterion variable by means of a logical combination of dichotomous predictors. *Mathématiques, Informatiques et Sciences Humaines*, 102:47–54.

Van Mechelen, I., Bock, H.-H., and De Boeck, P. (2004). Two-mode clustering methods: a structural overview. *Statistical Methods in Medical Research*, 13:363–394.

Van Mechelen, I., De Boeck, R., and Rosenberg, S. (1995). The conjunctive model of hierarchical classes. *Psychometrika*, 60:505–521.

Van Mechelen, I., Lombardi, L., and Ceulemans, E. (2007). Hierarchical classes modeling of rating data. *Psychometrika*, 72:475–88.

Van Mechelen, I. and Smilde, A. K. (2010). A generic linked-mode decomposition model for data fusion. *Chemometrics and Intelligent Laboratory Systems*, 104:83–94.

Vasilescu, M. A. O. (2002). Human motion signatures: Analysis, synthesis, recognition. In *ICPR 2002: Proceedings of the 16th International Conference on Pattern Recognition, IEEE Computer Society Press.*

Vasilescu, M. A. O. and Terzopoulos, D. (2002). Multilinear analysis of image ensembles: Tensorfaces. In *ECCV 2002:Proceedings of the 7th European Conference on Computer Vision, Lecture Notes in Comput. Sci. 2350, Springer, 447-460.*

Vasilescu, M. A. O. and Terzopoulos, D. (2003). Multilinear subspace analysis of image ensembles. In *CVPR 2003: Proceedings of the 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, IEEE Computer Society Press.*

Vasilescu, M. A. O. and Terzopoulos, D. (2004). Tensortextures: Multilinear image-based rendering. *ACM Trans. Graphics*, 23:336–342.

Vervloet, M., Van Deun, K., Van den Noortgate, W., and Ceulemans, E. (2013). On the selection of the weighting parameter value in principal covariates regression. *Chemometrics and Intelligent Laboratory Systems*, 123:36–43.

Vicente-Villardón, J. L., Galindo, P., and Blázquez, A. (2006). *Multiple Correspondence Analysis and Related Methods.*, chapter Logistic biplots. Chapman & Hall, New York.

Vlasic, D., Brand, M., Pfister, H., and Popovi, J. (2005). Face transfer with multilinear models. *ACM Trans. Graphics*, 24:426–433.

Vos, M. D., Lathauwer, L. D., Vanrumste, B., Huffel, S. V., and Paesschen, W. V. (2007a). Canonical decomposition of ictal scalp eeg and accurate source localisation: Principles and simulation study. *Comput. Intelligence Neurosci*, pages 1–8.

Vos, M. D., Vergult, A., Lathauwer, L. D., Clercq, W. D., Huffel, S. V., Dupont, P., Palmini, A., and Paesschen, W. V. (2007b). Canonical decomposition of ictal scalp eeg reliably detects the seizure onset zone. *NeuroImage*, 37:844–854.

Wang, H. and Ahuja, N. (2003). Facial expression decomposition. In *ICCV 2003: Proceedings of the9th IEEE International Conference on Computer Vision.*

Weesie, J. and Houwelingen, H. V. (1983). *GEPCAMP user's manual (first draft)*. Utrecht, The Netherlands: Institute of Mathematical Satistics, State University of Utrecht.

Wilderjans, T. F. and Ceulemans, E. (2013). Clusterwise parafac to identify heterogeneity in three-way data. *Chemometrics and Intelligent Laboratory Systems*, 129:87–97.

Wilderjans, T. F., Ceulemans, E., Kiers, H. A. L., and Meers, K. (2009a). The lmpca program: A graphical user interface for fitting the linked-mode parafac-pca model to coupled real-valued data. *Behaviour Research Methods*, 41 (4):1073–1082.

Wilderjans, T. F., Ceulemans, E., and Meers, K. (2013). A generic convex-hull-based model selection method. *Behavior Research Methods*, 45:1–15.

Wilderjans, T. F., Ceulemans, E., and Van Mechelen, I. (2005). The chic model: A global model for coupled binary data. Technical report, Katholieke Universiteit Leuven.

Wilderjans, T. F., Ceulemans, E., and Van Mechelen, I. (2008). The chic model: A global model for coupled binary data. *Psychometrika*, 73:729–751.

Wilderjans, T. F., Ceulemans, E., and Van Mechelen, I. (2009b). Simultaneous analysis of coupled data blocks differing in size: A comparison of two weighting schemes. *Computational Statistics & Data Analysis*, 53:1086–1098.

Wilderjans, T. F., Ceulemans, E., and Van Mechelen, I. (2012). The simclas model: Simultaneous analysis of coupled binary data matrices with noise heterogeneity between and within data blocks. *Psychometrika*, 77:724–740.

Yan, W., Hunt, L. A., Sheng, Q., and Szlavnics, Z. (2000). Cultivar evaluation and mega-environment investigation based on gge biplot. *Crop Sci*, 40:597–605.

Yan, W. and Kang, M. S. (2003). *GGE Biplot Analysis: A Graphical Tool for Breeders, Geneticists, and Agronomists*. CRC Press, Boca Raton, FL, USA.

Yan, W., Kang, M. S., Ma, B., Woods, S., and Cornelius., P. (2007). Gge biplot vs. ammi analysis of genotype-by-environment data. *Crop Sci.*, 47:643–655.

Yan, W. and Tinker., N. A. (2006). Biplot analysis of multi-environment trial data: Principles and applications. *Can. J. Plant Sci.*, 86:623–645.

Yan, W. and Tinker, N. A. (2006). Biplot analysis of multi-environment trial data: Principles and applications. *Can. J. Plant Sci.*, 86:623–645.

Yates, F. (1933). The analysis of replicate experiments when the field results are incomplete. *Empire Journal of Experimental Agriculture*, 1:129–142.

Zander, E. and Matthies, H. G. (2007). Tensor product methods for stochastic problems. *PAMM Proc. Appl. Math. Mech.*, 7:2040067–2040068.

Zijlstra, B. J. H. and Kiers, H. A. L. (2002). Degenerate solutions obtained from several variants of factor analysis. *Journal of Chernometrics*, 16:596–605.

ANEXOS

INVENTARIO DE ESTRÉS ACADEMICO

1. Edad:

2. Sexo:

- ☐ Masculino
- ☐ Femenino

3. Facultad

4. Grado

5. Señala con qué frecuencia te inquietó **la realización de un examen:**

- ☐ Nunca
- ☐ Rara vez
- ☐ Algunas veces
- ☐ Casi siempre
- ☐ Siempre

Las reacciones que siento son:

	Nunca	**Rara vez**	**Algunas veces**	**Casi siempre**	**Siempre**
Inquietud					
El corazón me late muy rápido y/o me falta aire y la respiración es agitada					
Realizo movimientos repetitivos con alguna parte de mi cuerpo, me quedo paralizado o mis movimientos son torpes					
Siento miedo					
Siento molestias en el estomago					
Fumo, como o bebo demasiado					
Tengo pensamientos o sentimientos negativos					
Me tiemblan las manos o las piernas					
Me cuesta expresarme verbalmente o a veces tartamudeo					
Me siento inseguro de mi mismo					
Se me seca la boca y tengo dificultades para tragar					
Siento ganas de llorar					

6. Señala con qué frecuencia te inquietó la **exposición de trabajos en clase:**

- ☐ Nunca
- ☐ Rara vez
- ☐ Algunas veces
- ☐ Casi siempre
- ☐ Siempre

Las reacciones que siento son:

	Nunca	**Rara vez**	**Algunas veces**	**Casi siempre**	**Siempre**
Inquietud					
El corazón me late muy rápido y/o me falta aire y la respiración es agitada					
Realizo movimientos repetitivos con alguna parte de mi cuerpo, me quedo paralizado o mis movimientos son torpes					
Siento miedo					
Siento molestias en el estomago					
Fumo, como o bebo demasiado					
Tengo pensamientos o sentimientos negativos					
Me tiemblan las manos o las piernas					
Me cuesta expresarme verbalmente o a veces tartamudeo					
Me siento inseguro de mi mismo					
Se me seca la boca y tengo dificultades para tragar					
Siento ganas de llorar					

7. Señala con qué frecuencia te inquietó la **intervención en el aula (responder a una pregunta del profesor, realizar preguntas, participar en coloquios, etc):**

☐ Nunca ☐ Casi siempre
☐ Rara vez ☐ Siempre
☐ Algunas veces

Las reacciones que siento son:

	Nunca	**Rara vez**	**Algunas veces**	**Casi siempre**	**Siempre**
Inquietud					
El corazón me late muy rápido y/o me falta aire y la respiración es agitada					
Realizo movimientos repetitivos con alguna parte de mi cuerpo, me quedo paralizado o mis movimientos son torpes					
Siento miedo					
Siento molestias en el estomago					
Fumo, como o bebo demasiado					
Tengo pensamientos o sentimientos negativos					
Me tiemblan las manos o las piernas					
Me cuesta expresarme verbalmente o a veces tartamudeo					
Me siento inseguro de mi mismo					
Se me seca la boca y tengo dificultades para tragar					
Siento ganas de llorar					

8. Señala con qué frecuencia te inquietó **subir al despacho del profesor en horas de tutorías:**

☐ Nunca ☐ Casi siempre
☐ Rara vez ☐ Siempre
☐ Algunas veces

Las reacciones que siento son:

	Nunca	**Rara vez**	**Algunas veces**	**Casi siempre**	**Siempre**
Inquietud					
El corazón me late muy rápido y/o me falta aire y la respiración es agitada					
Realizo movimientos repetitivos con alguna parte de mi cuerpo, me quedo paralizado o mis movimientos son torpes					
Siento miedo					
Siento molestias en el estomago					
Fumo, como o bebo demasiado					
Tengo pensamientos o sentimientos negativos					
Me tiemblan las manos o las piernas					
Me cuesta expresarme verbalmente o a veces tartamudeo					
Me siento inseguro de mi mismo					
Se me seca la boca y tengo dificultades para tragar					
Siento ganas de llorar					

9. Señala con qué frecuencia te inquietó la **sobrecarga académica (excesivo número de créditos, trabajos obligatorios, etc):**

☐ Nunca ☐ Casi siempre
☐ Rara vez ☐ Siempre
☐ Algunas veces

Las reacciones que siento son:

	Nunca	Rara vez	Algunas veces	Casi siempre	Siempre
Inquietud					
El corazón me late muy rápido y/o me falta aire y la respiración es agitada					
Realizo movimientos repetitivos con alguna parte de mi cuerpo, me quedo paralizado o mis movimientos son torpes					
Siento miedo					
Siento molestias en el estomago					
Fumo, como o bebo demasiado					
Tengo pensamientos o sentimientos negativos					
Me tiemblan las manos o las piernas					
Me cuesta expresarme verbalmente o a veces tartamudeo					
Me siento inseguro de mi mismo					
Se me seca la boca y tengo dificultades para tragar					
Siento ganas de llorar					

10. Señala con qué frecuencia te inquietó la **masificación de las aulas**

☐ Nunca ☐ Casi siempre
☐ Rara vez ☐ Siempre
☐ Algunas veces

Las reacciones que siento son:

	Nunca	Rara vez	Algunas veces	Casi siempre	Siempre
Inquietud					
El corazón me late muy rápido y/o me falta aire y la respiración es agitada					
Realizo movimientos repetitivos con alguna parte de mi cuerpo, me quedo paralizado o mis movimientos son torpes					
Siento miedo					
Siento molestias en el estomago					
Fumo, como o bebo demasiado					
Tengo pensamientos o sentimientos negativos					
Me tiemblan las manos o las piernas					
Me cuesta expresarme verbalmente o a veces tartamudeo					
Me siento inseguro de mi mismo					
Se me seca la boca y tengo dificultades para tragar					
Siento ganas de llorar					

11. Señala con qué frecuencia te inquietó la **falta de tiempo para poder cumplir con las actividades académicas:**

☐ Nunca ☐ Casi siempre
☐ Rara vez ☐ Siempre
☐ Algunas veces

Las reacciones que siento son:

	Nunca	Rara vez	Algunas veces	Casi siempre	Siempre
Inquietud					
El corazón me late muy rápido y/o me falta aire y la respiración es agitada					

	Nunca	Rara Vez	Algunas Veces	Casi Siempre	Siempre
Realizo movimientos repetitivos con alguna parte de mi cuerpo, me quedo paralizado o mis movimientos son torpes					
Siento miedo					
Siento molestias en el estomago					
Fumo, como o bebo demasiado					
Tengo pensamientos o sentimientos negativos					
Me tiemblan las manos o las piernas					
Me cuesta expresarme verbalmente o a veces tartamudeo					
Me siento inseguro de mi mismo					
Se me seca la boca y tengo dificultades para tragar					
Siento ganas de llorar					

12. Señala con qué frecuencia te inquietó **competitividad entre los compañeros:**

☐ Nunca ☐ Casi siempre
☐ Rara vez ☐ Siempre
☐ Algunas veces

Las reacciones que siento son:

	Nunca	Rara vez	Algunas veces	Casi siempre	Siempre
Inquietud					
El corazón me late muy rápido y/o me falta aire y la respiración es agitada					
Realizo movimientos repetitivos con alguna parte de mi cuerpo, me quedo paralizado o mis movimientos son torpes					
Siento miedo					
Siento molestias en el estomago					
Fumo, como o bebo demasiado					
Tengo pensamientos o sentimientos negativos					
Me tiemblan las manos o las piernas					
Me cuesta expresarme verbalmente o a veces tartamudeo					
Me siento inseguro de mi mismo					
Se me seca la boca y tengo dificultades para tragar					
Siento ganas de llorar					

13. Señala con qué frecuencia te inquietó **la realización de trabajos obligatorios para aprobar la asignatura (búsqueda de material necesario, redactar el trabajo, etc):**

☐ Nunca ☐ Casi siempre
☐ Rara vez ☐ Siempre
☐ Algunas veces

Las reacciones que siento son:

	Nunca	Rara vez	Algunas veces	Casi siempre	Siempre
Inquietud					
El corazón me late muy rápido y/o me falta aire y la respiración es agitada					
Realizo movimientos repetitivos con alguna parte de mi cuerpo, me quedo paralizado o mis movimientos son torpes					
Siento miedo					
Siento molestias en el estomago					
Fumo, como o bebo demasiado					
Tengo pensamientos o sentimientos negativos					

	Nunca	Rara vez	Algunas veces	Casi siempre	Siempre
Me cuesta expresarme verbalmente o a veces tartamudeo					
Me siento inseguro de mi mismo					
Se me seca la boca y tengo dificultades para tragar					
Siento ganas de llorar					

14. Señala con qué frecuencia te inquietó **la tarea de estudio:**

☐ Nunca ☐ Casi siempre
☐ Rara vez ☐ Siempre
☐ Algunas veces

Las reacciones que siento son:

	Nunca	Rara vez	Algunas veces	Casi siempre	Siempre
Inquietud					
El corazón me late muy rápido y/o me falta aire y la respiración es agitada					
Realizo movimientos repetitivos con alguna parte de mi cuerpo, me quedo paralizado o mis movimientos son torpes					
Siento miedo					
Siento molestias en el estomago					
Fumo, como o bebo demasiado					
Tengo pensamientos o sentimientos negativos					
Me tiemblan las manos o las piernas					
Me cuesta expresarme verbalmente o a veces tartamudeo					
Me siento inseguro de mi mismo					
Se me seca la boca y tengo dificultades para tragar					
Siento ganas de llorar					

15. Señala con qué frecuencia te inquietó **trabajar en grupo:**

☐ Nunca ☐ Casi siempre
☐ Rara vez ☐ Siempre
☐ Algunas veces

Las reacciones que siento son:

	Nunca	Rara vez	Algunas veces	Casi siempre	Siempre
Inquietud					
El corazón me late muy rápido y/o me falta aire y la respiración es agitada					
Realizo movimientos repetitivos con alguna parte de mi cuerpo, me quedo paralizado o mis movimientos son torpes					
Siento miedo					
Siento molestias en el estomago					
Fumo, como o bebo demasiado					
Tengo pensamientos o sentimientos negativos					
Me tiemblan las manos o las piernas					
Me cuesta expresarme verbalmente o a veces tartamudeo					
Me siento inseguro de mi mismo					
Se me seca la boca y tengo dificultades para tragar					
Siento ganas de llorar					