UNIVERSIDAD DE SALAMANCA

DEPARTAMENTO DE ESTADÍSTICA

DOCTORADO EN ESTADÍSTICA MULTIVARIANTE APLICADA

TESIS DOCTORAL

"ANÁLISIS DE COMPONENTES PRINCIPALES DISJUNTAS POR MEDIO DE OPTIMIZACIÓN POR ENJAMBRE DE PARTÍCULAS Y SUS APLICACIONES"

AUTOR: JOHN ALEX RAMÍREZ FIGUEROA

DIRECTORAS: MARÍA PURIFICACIÓN GALINDO VILLARDÓN

ANA BELÉN NIETO LIBRERO

2021

ANÁLISIS DE COMPONENTES PRINCIPALES DISJUNTAS POR MEDIO DE OPTIMIZACIÓN POR ENJAMBRE DE PARTÍCULAS Y SUS APLICACIONES

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

John Alex Ramírez Figueroa

Salamanca, 2021

UNIVERSIDAD DE SALAMANCA

DEPARTAMENTO DE ESTADÍSTICA

MARÍA PURIFICACIÓN GALINDO VILLARDÓN

Catedrática de la Universidad de Salamanca

Y

ANA BELÉN NIETO LIBRERO

Profesora Ayudante Doctor de la Universidad de Salamanca

CERTIFICAN:

Que Don John Alex Ramírez Figueroa 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 COMPONENTES PRINCIPALES DISJUNTAS POR MEDIO DE OPTIMIZACIÓN POR ENJAMBRE DE PARTÍCULAS Y SUS APLICACIONES** y para que conste, firman el presente certificado en Salamanca, a 7 de diciembre del 2021.

AGRADECIMIENTOS

Al término de este trabajo y después de haber recorrido un camino de aprendizaje tanto académico como personal quiero agradecer, ante todo, la invaluable ayuda de mi directora de tesis la Dra. María Purificación Galindo Villardón, y mi cotutora la Dra. Ana Belén Nieto Librero, por sus enseñanzas y paciencia. Sin sus consejos y compromiso este trabajo no habría sido posible.

A mi esposa e hija, que en todo momento estuvieron dándome ánimo y amor. Sin su ejemplo y sacrificio me habría rendido.

A mi madre, que a la distancia siempre me envió sus bendiciones.

A mi hermano, que siempre me apoyó en todo sentido.

A mi amigo y colega Carlos.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

Notaciones

$\mathbf{X}$	Matriz de tamaño $I \times J$
$\mathbf{A}$	Matriz de tamaño $I \times Q$
$\mathbf{B}$	Matriz de tamaño $J \times Q$
$F(\mathbf{A}, \mathbf{B})$	Función objetivo
$\|\mathbf{X}\|$	Norma de Frobenius de la matriz $\mathbf{X}$
v_{jq}	Variables binarias
$\mathbf{V}_0$	Matriz de variables binarias
$\mathbf{W}_{kq}$	Submatrices de $\mathbf{X}$
$\mathbf{B}_k$	Matriz de cargas
$\mathbf{A}_k$	Matriz de scores
$\mathbf{v}_i^t$	Velocidad de la partícula i en la etapa t
$\mathbf{x}_i^t$	Posición de la partícula i al tiempo t
ϕ_1	Ratio de aprendizaje individual
ϕ_2	Ratio de aprendizaje social
ρ_1, ρ_2	Valores aleatorios en el intervalo $[0,1]$
$newVel_p$	Matriz de velocidades de la partícula p
Vel_p	Matriz de velocidades de la partícula p en el intervalo $[-1,1]$
L	Operador que aplica la función sigmoidea
$tempB_p$	Posición temporal de la partícula p

$bestFit_p$	Mejor ajuste de la partícula p
σ_j^2	Varianza de la variable j
$tr(\mathbf{S})$	Traza de la matriz **S**
$\mathbf{\Sigma}$	Varianza de la matriz **A**
φ	Operador que genera una matriz aleatoria con estructura disjunta
$\langle \cdot, \cdot \rangle$	Producto interno
$\|v\|$	Norma del vector v
H	Espacio de Hilbert
$L^2(T)$	Espacio de las funciones de cuadrado integrable
$\mathrm{Ker}(L)$	Núcleo del operador L
$\eta(L)$	Nulidad del operador L
$\mathrm{Im}\ L$	Imagen del operador L
$rg(L)$	Rango del operador L
$\mathcal{B}(\mathbf{V}, \mathbb{R})$	Espacio vectorial de los funcionales lineales acotados
$\mathcal{F}(f)$	Operador integral de f
$\mathcal{F}^*$	Operador adjunto de $\mathcal{F}$
$\mathcal{B}_{HS}(H_1, H_2)$	Espacio de todas las transformaciones de Hilbert-Schmidt de H_1 en H_2
$\|\mathcal{F}\|_{HS}$	Norma de Hilbert-Schmidt $\mathcal{F}$
$\|\mathcal{F}\|_{TR}$	Norma traza de $\mathcal{F}$

$K(s,t)$	Núcleo del operador $\mathcal{F}$
$\overline{C}(s,t)$	Estimador muestral de la covarianza
$\{\psi_1,\psi_2,\ldots,\psi_K\}$	Base ortonormal de funciones
$\boldsymbol{\Psi}$	Vector que contiene a la base $\{\psi_1,\psi_2,\ldots,\psi_K\}$
$\mathbf{G}$	Matriz de Gram de la base $\{\psi_1,\psi_2,\ldots,\psi_K\}$
$S(t)$	Spline
$\psi_{i,k}$	Elemento de la base B-spline

RESUMEN

En el presente trabajo se divide en cuatro capítulos. En los dos primeros se propone una alternativa al método disjoint principal component analysis, que consiste en un análisis de componentes principales con restricciones y permite determinar componentes disjuntas, que son combinaciones lineales de subconjuntos disjuntos de las variables consideradas en el problema. El nuevo método propuesto se denomina constrained binary optimization by particle swarm disjoint principal component analysis, debido a que está basado en la optimización por enjambre de partículas, notado por CBPSO DC**.** El nuevo método usa una optimización estocástica diseñada para encontrar soluciones de alta calidad, en situaciones de alta complejidad computacional. El algoritmo del nuevo método parte generando aleatoriamente una población de partículas que iterativamente evolucionan hasta alcanzar el óptimo global, que en este caso está dado en función de las componentes disjuntas. Se proporcionan resultados numéricos que confirman la calidad de las soluciones obtenidas por el nuevo método.

En el tercer capítulo se realiza una adaptación de la optimización por enjambre de partículas al análisis HJ Biplot. Este nuevo método recibe el nombre de PSO DHJ Biplot. Además, se realiza una aplicación a datos sobre la evolución de la pandemia COVID-19.

En el capítulo cuatro se presentan los contenidos teóricos necesarios para abordar la estadística de datos funcionales. Se realiza una exposición de los operadores lineales sobre espacios de Hilbert. La función de covarianzas de un conjunto de datos funcionales es un ejemplo de este tipo de operadores. A continuación, se enuncia el teorema de Mercer y el teorema de Karhunen-Loève que son los dos pilares en que se sustenta el análisis de componentes principales funcionales. Se definen las componentes principales funcionales y se detalla su forma de cálculo. Se aplica el algoritmo CBPSO DC para obtener las denominadas componentes principales funcionales disjuntas.

Este documento se organiza de la siguiente manera:

INTRODUCCIÓN

CAPÍTULO UNO: ANÁLISIS DE COMPONENTES PRINCIPALES CLÁSICO

Se presenta el desarrollo teórico del Análisis de Componentes Principales Clásico, desde un punto de vista algebraico estadístico en dimensión finita. Esta presentación es necesaria para entender cómo se va a modificar el método clásico en presencia de restricciones como las que se utilizan en las componentes disjuntas, y también para hacer una generalización a espacios funcionales que tienen dimensión infinita.

CAPÍTULO DOS: METODOLOGÍA DEL ANÁLISIS DE COMPONENTES PRINCIPALES DISJUNTAS

En este capítulo se realiza una descripción pormenorizada del método de obtención de las componentes principales disjuntas de la forma clásica. A la vez que se presenta un nuevo algoritmo de cálculo, en el que se aplica la denominada optimización por enjambre de partículas. A este nuevo algoritmo se lo denomina CBPSO DC. Para finalizar se realizan aplicaciones del nuevo algoritmo a diferentes problemas con datos reales.

CAPÍTULO TRES: ANÁLISIS HJ BIPLOT DISJUNTO MEDIANTE OPTIMIZACIÓN POR ENJAMBRE DE PARTÍCULAS

El algoritmo desarrollado en el capítulo dos, se adapta para permitir realizar el análisis HJ Biplot disjunto mediante la optimización por enjambre de partículas. Para lo cual se añade al algoritmo CBPSO DC el cálculo de la matriz de valores singulares en cada iteración del algoritmo CBPSO DC. Se realiza una aplicación a datos.

CAPÍTULO CUATRO: ANÁLISIS DE COMPONENTES PRINCIPALES FUNCIONALES DISJUNTAS

Se exhibe un algoritmo para simular datos que tengan una estructura cuyas componentes principales tengan naturaleza disjunta, para luego comprobar si el algoritmo presentado en el capítulo tres detecta esta estructura factorial. Luego se hacen aplicaciones a tres conjuntos de datos reales. Se presenta la teoría necesaria para entender el desarrollo del Análisis de Componentes Principales Funcionales. Para ello se realiza una revisión de los principales resultados teóricos tanto de las nociones de norma y distancia y de los espacios de Hilbert, que es el espacio vectorial donde se encuentra los objetos denominados datos funcionales. Se revisan los principales resultados que dan soporte a la descomposición espectral cuando se disponen de datos funcionales. Los funcionales lineales, que son funciones entre dos espacios vectoriales juegan el mismo papel que matrices de varianzas-covarianzas. Se presenta la generalización del método de las Componentes Principales Clásico al ambiente funcional. Luego se presenta la propuesta de cálculo de las componentes principales disjuntas mediante optimización por enjambre de partículas. Se presenta un ejemplo de aplicación.

INTRODUCCIÓN

La estadística multivariante es un área de la ciencia en rápido crecimiento y desarrollo. Sus distintos procedimientos permiten determinar la existencia de estructuras latentes en una masa de datos, estructuras que pueden ser indetectables a través de métodos univariantes (MacGregor et al., 2015). Posibilitan descubrir patrones, agrupaciones y tendencias, caracterizarlos mediante relaciones existentes entre los objetos o individuos y variables, viabilizando la construcción de modelos de comportamiento, de asociación, de forma, dentro de la estructura global permitiendo al investigador una mejor interpretación del evento analizado (Camacho et al., 2015).

Uno de los métodos multivariantes clásico, para reducción de dimensión, es el denominado Análisis de Componentes Principales (PCA por sus siglas en inglés), desarrollado por Pearson (Pearson, 1901) y por Hotelling (Hotelling, 1933) de forma independiente en las primeras décadas del siglo pasado. Con el desarrollo de los métodos electrónicos de cálculo cobró gran importancia y hoy es el método multivariante no supervisado utilizado como primera línea en la investigación.

Una de los desafíos que plantea PCA es la interpretación de los resultados. Esto es, el significado de los valores de las coordenadas de los vectores directores de los nuevos ejes proporcionados por el análisis de componentes principales, denominados cargas (o loadings). Existen diversos criterios para determinar la importancia de las variables en cada uno de los ejes principales, ver (I. T. Jolliffe, 1972) y (I. T. Jolliffe, 1973). Hay investigadores que consideran que las variables más importantes son aquellas que tienen mayor valor absoluto. Otros eligen un umbral, por lo general 0.5. Las variables cuyos valores absolutos superan el umbral fijado se consideran que son las que caracterizan el eje principal. El problema se vuelve complicado cuando los valores absolutos de las cargas tienen poca variabilidad, esto es, son bastante similares. En esta situación el PCA se puede convertir en un análisis irrelevante en cuanto a la interpretación de los nuevos ejes principales.

Existen varias estrategias implementadas para paliar este inconveniente. Un resumen exhaustivo es presentado en el capítulo 2. Entre los principales métodos tenemos dos: Las componentes principales dispersas (Sparse PCA) y las componentes principales disjuntas

(Disjoint PCA). El núcleo de este capítulo es un nuevo método basado en la optimización por enjambre de partículas (PSO) (Kennedy & Eberhart, 1995) y (Eberhart & Yuhui, 2001) y en Disjoint PCA (Nieto-Librero, 2015). Este método original presenta un nuevo paradigma en cuanto a los métodos de optimización utilizados en el análisis multivariante, puesto que reemplaza al clásico algoritmo de los mínimos cuadrados alternantes (ALS) por un algoritmo evolutivo que impide caer en óptimos relativos encontrando el óptimo global con una frecuencia superior al ALS. Especialmente con matrices de datos de gran dimensión, tal como se muestra en los ejemplos del capítulo 2. En los ejemplos en el mostrados en el documento alcanza la solución óptima el 100% de las veces. Este algoritmo se denomina CBPSO DC.

En el capítulo 3 se extiende el algoritmo CBPSO DC al análisis HJ Biplot (Galindo, 1986), obteniendo un HJ Biplot cuyas componentes tienen estructura disjunta. Este algoritmo se ha denominado Disjoint HJ Biplot. Se realiza una aplicación a datos sobre la evolución de la pandemia COVID-19 en Ecuador para demostrar su utilidad.

En el capítulo 4 se realiza otra aplicación-extensión del algoritmo CBPSO DC al campo funcional. Puesto que el análisis de componentes principales funcionales trata con objetos que no son vectores de dimensión finita sino funciones definidas sobre espacios de dimensión infinita se hace necesario una introducción teórica para entender en primer lugar a los objetos matemáticos con los que se trata en el ámbito funcional (Hsing & Eubank, 2013), y en segundo lugar explicar el proceso matemático para convertirlos en datos discretos para su tratamiento matricial (J. O. Ramsay & Silverman, 2005) y (Di Salvo et al., 2015). Para comprobar la efectividad del nuevo método se realiza una aplicación a datos de temperaturas de estaciones climatológicas de Canadá.

Capítulo 1

ANÁLISIS DE COMPONENTES PRINCIPALES CLÁSICO

En este capítulo se realiza una revisión de la teoría del Análisis de Componentes Principales. Para ello se recurre a la teoría algebraica de valores y vectores propios, además de los multiplicadores de Lagrange. Puesto que en este documento se van a presentar varios métodos para el cálculo de las denominadas componentes principales disjuntas, para diferenciar lo vamos a denominar Análisis de Componentes Principales Clásico (PCA clásico).

El capítulo está distribuido de la siguiente manera:

1.1 Introducción.

1.2 Análisis de Componentes Principales Matricial.

1.3 Descomposición en Valores Singulares.

1.1 Introducción

El PCA clásico se utiliza ampliamente en los estudios multivariantes como un método de reducción de la dimensionalidad de un conjunto de datos. Para ello se recurre a la representación espectral de la matriz de varianzas-covarianzas del conjunto de datos analizado. Las nuevas variables, que son combinación lineal de las variables originales, se denominan componentes principales, y corresponden a los vectores propios normalizados de la matriz de varianzas-covarianzas. El objetivo de este capítulo es el de proporcionar una síntesis del Análisis de Componentes Principales de una matriz de datos, notada por $\mathbf{X}$, y de la descomposición en valores singulares (SVD)

1.2 Análisis de componentes Principales Matricial.

Los valores y vectores propios están en el núcleo de la estadística multivariante, de la ciencia de datos, aprendizaje automático y física, por citar algunas ramas del conocimiento.

Muchos problemas conducen a la expresión $\mathbf{A}v = \lambda v$, es decir, a encontrar un vector que al ser multiplicado por una matriz cuadrada $\mathbf{A}$ no altere su dirección, máxime su sentido.

Definición 1.1.- Sea una matriz cuadrada $\mathbf{A} \in \mathbb{C}^{n\times n}$. El número real o complejo λ se denomina valor propio de $\mathbf{A}$ si existe un vector diferente de cero $v \in \mathbb{C}^n$ tal que:

$$\mathbf{A}v = = \lambda v \tag{1.2.1}$$

El vector v se denomina vector propio de $\mathbf{A}$ correspondiente al valor propio λ.

Los resultados fundamentales sobre valores y vectores y propios que nos interesan en este trabajo son los siguientes:

Teorema 1.1.- Toda matriz $\mathbf{A}$ simétrica real de tamaño $n\times n$ es diagonalizable ortogonalmente.

Teorema 1.2.- Si $\mathbf{A}$ es una matriz simétrica real de tamaño $n\times n$, entonces

a. Sus valores propios son positivos.
b. Los vectores propios correspondientes a valores propios diferentes son ortogonales.

Teorema 1.3.- (Teorema de descomposición espectral). Sea una matriz cuadrada simétrica real $\mathbf{A}$ de tamaño $n\times n$. Si $\lambda_1,\ldots,\lambda_n$ son los valores propios de $\mathbf{A}$ con correspondientes vectores propios $v_1,\ldots,v_n$, entonces:

$$\mathbf{A} = = \sum_{i=1}^{n} \lambda_i v_i^t \tag{1.2.2}$$

1.2.1 La metodología PCA

Se dispone de una muestra de *n* objetos o individuos y de *p* variables aleatorias centradas $X_1, X_2, \ldots, X_p$. Se desea obtener $k \leq p$ variables no correlacionadas $Z_1, Z_2, \ldots, Z_k$ que sean combinación lineal de las variables originales X_j de tal forma que expliquen la mayor parte de la variabilidad de las variables originales.

La primera componente principal, al igual que las restantes, se expresa como combinación lineal de las variables X_j de la forma:

$$Z_{1i} = u_{11}X_{1i} + u_{12}X_{2i} + \cdots + u_{1p}X_{pi} \quad \text{con } i = 1, 2, \ldots, n \tag{1.2.3}$$

Matricialmente:

$$\underbrace{\begin{pmatrix} Z_{11} \\ Z_{12} \\ \vdots \\ Z_{1n} \end{pmatrix}}_{\mathbf{Z}_1} = \underbrace{\begin{pmatrix} X_{11} & X_{21} & \cdots & X_{p1} \\ X_{12} & X_{22} & \cdots & X_{p1} \\ \vdots & \vdots & \cdots & \vdots \\ X_{1n} & X_{2n} & \cdots & X_{pp} \end{pmatrix}}_{\mathbf{X}} \underbrace{\begin{pmatrix} u_{11} \\ u_{12} \\ \vdots \\ u_{1p} \end{pmatrix}}_{\mathbf{u}_1} \tag{1.2.4}$$

Es decir, $\mathbf{Z}_1 = \mathbf{X}\mathbf{u}_1$, calculando su valor esperado:

$$\mathrm{E}[\mathbf{Z}_1] = \mathrm{E}[\mathbf{X}\mathbf{u}_1] = \mathrm{E}[\mathbf{X}]\mathbf{u}_1 = \mathbf{0}$$

La varianza $\mathbf{Z}_1$ está dada por:

$$Var[\mathbf{Z}_1] = \frac{1}{n}\sum_{i=1}^{n} Z_{1i}^2 = \frac{1}{n}\mathbf{Z}_1^{\dagger}\mathbf{Z}_1 = \mathbf{u}_1^{\dagger}\frac{\mathbf{X}^{\dagger}\mathbf{X}}{n}\mathbf{u}_1$$

Puesto que $\mathrm{Var}[X] = \dfrac{\mathbf{X}^{\dagger}\mathbf{X}}{n} = \mathbf{V}$, se tiene:

$$\mathrm{Var}\left[\mathbf{Z}_1\right] = \mathbf{u}^{\dagger}\mathbf{V}\mathbf{u} \tag{1.2.5}$$

Además,

$$\mathrm{tr}\left[\mathbf{V}\right] = \sigma_X^2 + \sigma_{X_2}^2 + \cdots + \sigma_{X_p}^2 \tag{1.2.6}$$

La primera componente principal es la combinación lineal (1.2.3) que tiene la máxima varianza y tal que $\|\mathbf{u}_1\| = 1$.

El problema de optimización consiste en determinar la combinación lineal $\mathbf{Z}_1$ que tenga la máxima varianza, sujeto a la restricción $\|\mathbf{u}_1\| = \mathbf{u}_1^{\dagger}\mathbf{u}_1 = 1$.

Para resolver este problema tomamos el lagrangiano:

$$L = \mathbf{u}_1^{\dagger}\mathbf{V}\mathbf{u}_1 - \lambda\left(\mathbf{u}_1^{\dagger}\mathbf{u}_1 - 1\right) \tag{1.2.7}$$

Derivando (1.2.7) e igualando a cero:

$$\frac{\partial L}{\partial \mathbf{u}_1} = 2\mathbf{V}\mathbf{u}_1 - 2\lambda\mathbf{u}_1 = \mathbf{0}$$

$$\mathbf{V}\mathbf{u}_1 = \lambda\mathbf{u}_1 \tag{1.2.8}$$

Lo que significa que $\mathbf{u}_1$ es el vector propio asociado al valor propio λ de la matriz $\mathbf{V}$.

$\mathbf{V}$ es una matriz simétrica semi-definida positiva por lo que tiene p valores propios λ_j ordenados en forma decreciente $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_p \geq 0$.

Si a la ecuación (1.2.8) la multiplicamos por $\mathbf{u}_1^{\dagger}$:

$$\underbrace{\mathbf{u}_1^{\dagger}\mathbf{V}\mathbf{u}_1}_{\mathrm{Var}\left[\mathbf{Z}_1\right]} = \lambda\mathbf{u}_1^{\dagger}\mathbf{u}_1$$

Como $\mathbf{u}_1$ es unitario tenemos $\text{Var}[\mathbf{Z}_1] = \lambda$, como deseamos que la varianza de $\mathbf{Z}_1$ sea máxima tomamos $\lambda = \lambda_1$ el mayor valor propio de $\mathbf{V}$.

Por tanto, los coeficientes de la combinación lineal $\mathbf{Z}_1$ están dados por los componentes del vector propio $\mathbf{u}_1$ asociado a λ_1:

$$\mathbf{Z}_1 = \mathbf{X}\mathbf{u}_1 \tag{1.2.9}$$

Para determinar la segunda componente principal realizamos un procedimiento análogo al seguido para calcular la primera componente principal:

$$Z_{2i} = u_{21}X_{1i} + u_{22}X_{2i} + \cdots + u_{2p}X_{pi} \quad \text{con } i = 1, 2, \ldots, n \tag{1.2.10}$$

En formato matricial: $\mathbf{Z}_2 = \mathbf{X}\mathbf{u}_2$, donde $\mathbf{u}_2 = (u_{21}, u_{22}, \ldots, u_{2p})^{\dagger}$.

Además:

$$\text{E}[\mathbf{Z}_2] = \mathbf{0} \text{ y } \text{Var}[\mathbf{Z}_2] = \mathbf{u}_2^{\dagger}\mathbf{V}\mathbf{u}_2 .$$

El problema de optimización consiste en determinar la variable aleatoria $\mathbf{Z}_2$ tal que su varianza sea máxima, no esté correlacionada con $\mathbf{Z}_1$ y tal que el vector $\mathbf{u}_2$ sea unitario.

Esto implica que:

$$\text{Cov}(\mathbf{Z}_1, \mathbf{Z}_2) = \text{Cov}(\mathbf{X}\mathbf{u}_1, \mathbf{X}\mathbf{u}_2) = \mathbf{0}$$

De donde:

$$\mathbf{u}_1^{\dagger}\,\text{Cov}(\mathbf{X}, \mathbf{X})\mathbf{u}_2 = \mathbf{0}$$
$$\mathbf{u}_1^{\dagger}\,\text{Var}[\mathbf{X}]\mathbf{u}_2 = \mathbf{0}$$

Por tanto:

$$\mathbf{u}_1^{\dagger}\mathbf{V}\mathbf{u}_2 = \mathbf{0} \tag{1.2.11}$$

Si a $\mathbf{V}\mathbf{u}_1 = \lambda_1 \mathbf{u}_1$ multiplicamos por $\mathbf{u}_2^\dagger$:

$$\mathbf{u}_2^\dagger \mathbf{V}\mathbf{u}_1 = \lambda_1 \mathbf{u}_2^\dagger \mathbf{u}_1$$

Por (1.2.11) se tiene que $\lambda_1 \mathbf{u}_2^\dagger \mathbf{u}_1 = 0$, es decir, $\mathbf{u}_2^\dagger \mathbf{u}_1 = 0$. En otras palabras, el hecho de que $\mathbf{Z}_1$ y $\mathbf{Z}_2$ no estén correlacionadas implica que los vectores $\mathbf{u}_1$ y $\mathbf{u}_2$ son ortogonales.

Así, el problema de optimización se puede expresar de la siguiente forma:

Maximixar $\mathrm{Var}[\mathbf{Z}_2] = \mathbf{u}_2^\dagger \mathbf{V}\mathbf{u}_2$, sujeta a las restricciones $\mathbf{u}_2^\dagger \mathbf{u}_2 = 1$ y $\mathbf{u}_1^\dagger \mathbf{V}\mathbf{u}_2 = \mathbf{0}$.

El lagrangiano correspondiente es:

$$L = \mathbf{u}_2^\dagger \mathbf{V}\mathbf{u}_2 - \mu\left(\mathbf{u}_1^\dagger \mathbf{V}\mathbf{u}_2\right) - \lambda\left(\mathbf{u}_2^\dagger \mathbf{u}_2 - 1\right)$$

Derivando respecto a $\mathbf{u}_2$: $\dfrac{\partial L}{\partial \mathbf{u}_2} = 2\mathbf{V}\mathbf{u}_2 - \mu\mathbf{V}\mathbf{u}_1 - 2\lambda\mathbf{u}_2 = \mathbf{0}$

$$\mathbf{V}\mathbf{u}_2 - \frac{1}{2}\mu\mathbf{V}\mathbf{u}_1 - \lambda\mathbf{u}_2 = \mathbf{0} \qquad (1.2.12)$$

Multiplicando (1.2.12) por $\mathbf{u}_1^\dagger$: $\underbrace{\mathbf{u}_1^\dagger \mathbf{V}\mathbf{u}_2}_{0} - \frac{1}{2}\mu\mathbf{u}_1^\dagger \mathbf{V}\mathbf{u}_1 - \lambda\underbrace{\mathbf{u}_1^\dagger \mathbf{u}_2}_{0} = \mathbf{0}$

De donde: $\mu\mathbf{u}_1^\dagger \mathbf{V}\mathbf{u}_1 = 0$, por (1.2.5) $\mathbf{u}_1^\dagger \mathbf{V}\mathbf{u}_1 \neq 0$, entonces se concluye que $\mu = 0$. Así, (1.2.12) se reduce a: $\mathbf{V}\mathbf{u}_2 - \lambda\mathbf{u}_2 = \mathbf{0}$, lo que implica:

$$\mathbf{V}\mathbf{u}_2 = \lambda\mathbf{u}_2$$

Es decir, $\mathbf{u}_2$ es el vector propio asociado al valor propio λ.

Como se requiere que $\mathrm{Var}[\mathbf{Z}_2]$ sea máxima, basta tomar el mayor valor propio de $\mathbf{V}$, puesto que λ_1 ya se tomó para la primera componente principal, se tiene que $\lambda = \lambda_2$:

$$\mathbf{V}\mathbf{u}_2 = \lambda_2\mathbf{u}_2 \tag{1.2.13}$$

Y por tanto:

$$\mathbf{Z}_2 = \mathbf{X}\mathbf{u}_2 \tag{1.2.14}$$

Se procede de la misma forma para las siguientes componentes principales:

$$\mathbf{V}\mathbf{u}_k = \lambda_k\mathbf{u}_k \quad \text{con} \quad k = 1,\ldots,p \tag{1.2.15}$$

$$\mathbf{Z}_k = \mathbf{X}\mathbf{u}_k \quad \text{con} \quad k = 1,2,\ldots,p \tag{1.2.16}$$

Multiplicando (1.2.15) por $\mathbf{u}_k^\dagger$ obtenemos la varianza de las componentes principales:

$$\underbrace{\mathbf{u}_k^\dagger\mathbf{V}\mathbf{u}_k}_{\mathrm{Var}(\mathbf{Z}_k)} = \lambda_k\,\mathbf{u}_k^\dagger\mathbf{u}_k$$

Lo que implica que:

$$\mathrm{Var}[\mathbf{Z}_k] = \lambda_k$$

Además:

$$\sum_{k=1}^{p}\mathrm{Var}[\mathbf{Z}_k] = \sum_{k=1}\lambda_k = \mathrm{tr}(\mathbf{V}) \tag{1.2.17}$$

Por otra parte, ya que $\mathbf{V}$ es simétrica existe una matriz ortogonal $\mathbf{P}$ tal que $\mathbf{P}^\dagger\mathbf{V}\mathbf{P} = \mathbf{D}$ con $\mathbf{D} = \mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_p)$. Si tomamos la traza se tiene:

$$\begin{aligned}\mathrm{tr}(\mathbf{D}) &= \mathrm{tr}(\mathbf{P}^\dagger\mathbf{V}\mathbf{P}) \\ &= \mathrm{tr}(\mathbf{V}\mathbf{P}\mathbf{P}^\dagger) \\ &= \mathrm{tr}(\mathbf{V})\end{aligned} \tag{1.2.18}$$

Reemplazando (1.2.17) en (1.2.18):

$$\sum_{k=1}^{p}\lambda_k = \sum_{k=1}^{p}\sigma_k^2 \tag{1.2.19}$$

1.3 Descomposición en Valores Singulares

Toda matriz cuadrada simétrica es diagonalizable ortogonalmente. ¿Qué sucede si la matriz no es cuadrada? En este caso se dispone de la denominada descomposición en valores singulares (SVD por sus siglas en inglés), teoría desarrollada por Beltrami, Jordan y Sylvester de forma independiente en la segunda mitad del siglo XIX.

Teorema 1.4.- Sea una matriz $\mathbf{A} \in \mathbb{C}^{m\times n}$, entonces existen dos matrices ortogonales $\mathbf{U} \in \mathbb{C}^{m\times m}$, $\mathbf{V} \in \mathbb{C}^{n\times n}$ y una matriz diagonal no negativa $\mathbf{\Lambda} \in \mathbb{R}^{m\times n}$ tales que:

$$\mathbf{A} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^T \tag{1.3.1}$$

La expresión (1.3.1) es la denominada SVD.

La demostración del teorema se puede consultar en (Yanai et al., 2011).

Los elementos de la diagonal de $\mathbf{\Lambda}$ se denominan valores singulares de $\mathbf{A}$.

Las columnas de $\mathbf{U}$ son los vectores singulares izquierdos de $\mathbf{A}$.

Las columnas de $\mathbf{V}$ son los vectores singulares derechos de $\mathbf{A}$.

Por otro lado:

$$\begin{aligned}\mathbf{A}\mathbf{A}^T &= \mathbf{U}\mathbf{\Lambda}\mathbf{V}^T(\mathbf{U}\mathbf{\Lambda}\mathbf{V}^T) = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^T\mathbf{V}\mathbf{\Lambda}^T\mathbf{U}^T \\ &= \mathbf{U}(\mathbf{\Lambda}\mathbf{\Lambda}^T)\mathbf{U}^T\end{aligned}$$

$$\begin{aligned}\mathbf{A}^T\mathbf{A} &= (\mathbf{U}\mathbf{\Lambda}\mathbf{V}^T)^T\mathbf{U}\mathbf{\Lambda}\mathbf{V}^T = \mathbf{V}\mathbf{\Lambda}^T\mathbf{U}^T\mathbf{U}\mathbf{\Lambda}\mathbf{V}^T \\ &= \mathbf{V}(\mathbf{\Lambda}^T\mathbf{\Lambda})\mathbf{V}^T\end{aligned}$$

Lo que indica que las columnas de **U** son los vectores propios de $\mathbf{A}\mathbf{A}^T$, y las columnas de **V** son los vectores propios de $\mathbf{A}^T\mathbf{A}$. Además, los valores propios de $\mathbf{A}\mathbf{A}^T$ y $\mathbf{A}^T\mathbf{A}$ (que son los mismos) son iguales a los valores singulares al cuadrado de **A**.

En el caso de que **A** sea una matriz con componentes reales, las matrices $\mathbf{A}\mathbf{A}^T$ y $\mathbf{A}^T\mathbf{A}$ son simétricas y por ende sus valores propios van a ser reales, al igual que sus vectores propios van a tener componentes reales.

Capítulo 2

METODOLOGÍA DEL ANÁLISIS DE COMPONENTES PRINCIPALES DISJUNTAS

En este capítulo se realiza una explicación del funcionamiento del método de optimización utilizado para obtener las denominadas componentes principales disjuntas mediante ALS. Este método se denominará DPCA (Disjoint Principal Component Analysis). Además, se expone la teoría de la Optimización por Enjambre de Partículas de manera general. Luego se muestra el algoritmo genérico correspondiente. A continuación, se presenta la implementación del PSO genérico a la optimización para el cálculo de las componentes principales disjuntas. Este algoritmo se denomina CBPSO DC (Constrained Binary Particle Swarm Optimization Disjoint Components). El método descrito calcula de manera directa la matriz $\mathbf{A}$ y a partir de ésta la matriz $\mathbf{B}$ sin recurrir a la matriz que contiene los valores propios. Estos valores propios se utilizan para determinar la varianza explicada por cada eje principal. En este caso, al no disponer de los valores singulares y por ende de los valores propios se propone un método para estimar estos porcentajes.

Para mostrar las bondades del algoritmo CBPSO DC y la calidad de las soluciones que se pueden encontrar se utilizan dos clases de ejemplos. La primera clase considera dos ejemplos simulados, para ello se diseñó un simulador a partir de variables aleatorias que están uniformemente distribuidas. Se utiliza el método CBPSO DC para detectar la presencia de factores latentes, y luego se realiza una aplicación en la validación de cuestionarios.

El capítulo está distribuido de la siguiente manera:

2.1 Introducción

2.2 La metodología del Análisis de Componentes Principales Disjuntas

2.3 Metodología de la Optimización por Enjambre de Partículas.

2.4 Ejemplos Numéricos

2.5 Aplicación a validación de Cuestionarios

2.6 Aplicación a datos de medio ambiente

2.1 Introducción

La mayoría de procedimientos multivariantes se fundamenta en la descomposición en valores singulares (SVD) de la matriz de datos a analizar (I. Jolliffe, 2002). Esta descomposición permite encontrar un sistema referencial ortogonal, cuyos ejes se denominan ejes principales o factoriales y corresponden a las direcciones de máxima varianza (Beaton et al., 2014). Estos nuevos ejes (sus vectores directores) son combinaciones lineales de las variables originales y como tal son variables latentes, no observables directamente.

A cada eje principal le corresponde un vector director y los valores de sus coordenadas o cargas permiten interpretar y nombrar a los ejes según el contexto de la investigación llevada a cabo (Hastie T et al., 2017). Tarea que puede convertirse en un problema si una gran cantidad de coordenadas tienen valores absolutos aproximadamente iguales, sin que existan cargas con valores absolutos claramente mayores a las demás que permitan caracterizar al eje correspondiente a la variable latente en cuestión. (Eriksson et al., 2006).

Si cada uno de los ejes principales fuese combinación lineal de unas pocas variables originales se facilitaría la interpretación de éstos dentro del contexto de la investigación. Se han elaborado varios métodos multivariantes para conseguir este objetivo. Así, podemos citar el método propuesto en (Vines, 2000) que consiste en obtener direcciones que se puedan representar por vectores de números enteros con varias de sus coordenadas iguales a cero. Las componentes principales dispersas o Sparse PCA (Zou et al., 2004) que buscan ejes factoriales con pocas cargas diferentes de cero o la descomposición CUR (Mahoney & Drineas, 2009) que consiste en representar a la matriz de datos como un producto de matrices de bajo rango, que permite expresarla en función de un reducido número de columnas (variables) y/o filas (individuos u objetos). Otra propuesta es la de las componentes disjuntas (Disjoint Components), que a más de representar a una componente como combinación lineal de unas pocas variables originales, éstas no aparecen en las otras componentes, de allí el nombre de disjuntas.

El inicio de la evolución de la metodología de las componentes disjuntas se puede rastrear en Vigneau & Qannari, 2003, donde se propone un método que consiste en realizar un análisis de conglomerados jerárquico para luego asociar una variable latente a cada conglomerado previamente obtenido. Estas variables latentes resultan más fáciles de interpretar que las componentes principales clásicas. El procedimiento apunta a encontrar grupos de variables en torno a variables sintéticas o latentes mediante la maximización de las covarianzas entre las variables originales y las latentes.

Vichi y Saporta, 2009 proporcionan un nuevo enfoque al trabajo de Vigneau y Qannari al determinar las variables latentes por medio del Análisis de Componentes Principales (Principal Component Analysis) (PCA). Así, se propone un método que consiste en aplicar de manera secuencial el Análisis de Conglomerados (Cluster Analysis) (CA) y el PCA. Se utiliza el CA para reducir el número de objetos, considerando los centroides, y PCA para reducir el número de variables al considerar las primeras componentes principales, buscando maximizar la desviación entre los conglomerados de las componentes en el espacio reducido a través de un algoritmo de mínimos cuadrados alternantes. La nueva metodología se llama Clustering and Disjoint Principal Component Analysis (CDPCA), porque se requiere encontrar componentes asociadas a clases de variables disjuntas. En (Macedo & Freitas, 2015) se propone un algoritmo de cálculo para la metodología CDPCA de Vichi y Saporta. También se tiene el Clustering and Disjoint HJ Biplot propuesto en (A. Nieto-Librero et al., 2017), donde se aplica el algoritmo propuesto en (Vichi & Saporta, 2009) al HJ Biplot en lugar de PCA.

Si se requieren componentes que faciliten la interpretación de los resultados, sin recurrir a la existencia de conglomerados para caracterizarlas, tenemos las denominadas componentes disjuntas (Disjoint Components). En (Ana Nieto-Librero, 2015) se presenta una técnica para realizar un biplot disjunto denominada Disjoint HJ Biplot y su algoritmo de cálculo está fundamentado en el método expuesto en (Vichi & Saporta, 2009) y en (Macedo & Freitas, 2015). Consiste basicamente de dos partes que se relacionan alternadamente: la primera es de construcción, se trata de obtener una matriz de particiones del conjunto de variables originales, y con estas particiones construir una matriz de cargas

y otra de scores, y a partir de éstas últimas se calcula una función de ajuste que busca minimizar el error de aproximación. La segunda parte consiste en un problema de optimización combinatoria que busca determinar la matriz de particiones que produzca el mejor ajuste posible. En (Ferrara et al., 2016) se presentan tres metodologías para obtener componentes disjuntas: Stepwise PCA, Restricted PCA y Disjoint PCA. La primera metodología consiste en realizar varios PCA iterados sobre Q componentes relevantes (valores propios mayores a uno) y eliminar en cada iteración a la variable que tenga la mayor carga en la última componente relevante. Este procedimiento se repite hasta obtener una sola componente, que va a definir un subconjunto de variables. Luego se repite el proceso sobre la matriz que contenga a las variables excluidas previamente, hasta que cada variable sea asociada a una sola componente. Restricted PCA consiste en detectar Q componentes relevantes para luego asignar una variable a la componente en la que tenga mayor peso. Los demás pesos para esta variable son igualados a cero. La tercera estrategia es Disjoint PCA y es una modificación del método propuesto en (Vichi & Saporta, 2009) y será utilizada para comparar con los resultados del algoritmo presentado en este trabajo. En adelante a este método lo vamos a denominar Disjoint PCA Clásico (Classic Disjoint PCA). En (Ferrara et al., 2018) se presenta una versión inferencial del Disjoint PCA basada en criterios de máxima verosimilitud.

El presente trabajo está inspirado en el método Disjoint PCA arriba mencionado, mostrando un enfoque diferente a la forma en que se realiza el proceso de optimización. Para ello se exhibe un nuevo paradigma en la optimización de problemas estadísticos apuntando a resolver el problema de hallar el mínimo de la función objetivo mediante un algoritmo de la clase de enjambre o cúmulo de partículas, de tipo binario con restricciones. También se presenta una propuesta alternativa para obtener la matriz de valores propios a partir de las inercias de cada uno de los nuevos ejes encontrados, y con ello la construcción de indicadores de los porcentajes de explicación de cada eje disjunto obtenido.

Para comprender la naturaleza de los cambios planteados a la forma de abordar y resolver el problema de optimización en el método propuesto hay que explicar con detalle como opera la técnica Classic Disjoint PCA.

En otro orden de ideas tenemos la Optimización por Enjambre de Partículas (PSO) una técnica emergente de la computación evolutiva desarrollada por James Kennedy y Russell Eberhart en 1995 (Kennedy & Eberhart, 1995), es un método de optimización estocástica inspirado en el comportamiento social de enjambres de insectos o bancos de peces. Para operar el algoritmo PSO es necesario definir una función objetivo a optimizar (maximizar o minimizar). Este algoritmo se inicia creando de forma aleatoria una población de partículas, que son las potenciales soluciones, y luego actúa de forma iterativa buscando el óptimo requerido en base a la actualización de la posición y velocidad de las partículas. PSO ha sido aplicado en muchas áreas del conocimiento tales como redes neuronales, sistemas de control, computación gráfica, robótica, minería de datos, optimización de funciones en espacios de alta dimensionalidad, etc. (Eberhart & Yuhui, 2001), (Alatas & Akin, 2008), (Vasile & Buiu, 2011).

En lo concerniente a las aplicaciones de PSO en el contexto del análisis multivariante podemos indicar: En (Voss, 2005) se expone una metodología para usar PCA con PSO en el que el sistema referencial se traslada junto al enjambre. Para determinar las direcciones de los ejes del sistema referencial en cada iteración se utiliza PCA. Los problemas de optimización con restricciones tienen espacios de búsqueda acotados contenidos dentro de un espacio vectorial en el que están definidas las variables utilizadas en el problema, puede ocurrir que el enjambre quede atrapado en la zona de soluciones no factibles del espacio vectorial. Para solventar este problema en (Chu et al., 2011) se presenta un método que utiliza PCA para el manejo de las cotas de la región factible del espacio. Bibo Ma & Haiyan Ji, 2012, utilizan PCA para reducir dimensiones y seleccionar las componentes principales más importantes para luego aplicar PSO. En forma parecida a la anterior Zhao, Lin, & Zhang, 2014 muestran un procedimiento que utiliza PCA para determinar direcciones eficientes para utilizar en PSO. Una aplicación a la regresión multivariante se tiene en (Peng et al., 2012), en este trabajo PSO se utiliza para seleccionar los componentes de frecuencia a través de transformaciones wavelets. Song et al., 2017 utilizan PSO para optimización global en presencia de discontinuidad.

Otro rubro de aplicación de PSO en el contexto multivariante es en análisis de conglomerados. Uno de trabajos pioneros se describe en (Van Der Merwe & Engelbrecht, 2003) en el cual se utiliza PSO para encontrar los centroides de un número específico de conglomerados. (Gajawada & Toshniwal, 2012) presentan el algoritmo PCPSO (Projected Clustering Particle Swarm Optimization) que consiste en hallar subespacios de los conglomerados presentes en el conjunto de datos. En (Esmin & Matwin, 2012) se da cuenta del algoritmo HPSOM (Hybrid Particle Swarm Optimization with Mutation) que utiliza PSO conjuntamente con la mutación de los algoritmos genéticos para determinar los centroides de un número de conglomerados especificado por el usuario . (Wang et al., 2018) nos muestran el algoritmo denominado CSPSO (Chaotic Starling Particle Swarm Optimization) que utiliza las respuestas colectivas de una bandada de estorninos para mejorar la búsqueda global del óptimo e implementa una aplicación logística caótica para evitar que el proceso de búsqueda termine prematuramente.

En los artículos citados anteriormente, a excepción de los trabajos sobre análisis de conglomerados, los métodos multivariantes se utilizan dentro de PSO para mejorar su funcionamiento o se los utiliza en etapas previas o posteriores a la aplicación de PSO. En el presente trabajo se hace una propuesta diferente: la utilización de PSO dentro de los métodos multivariantes, específicamente en la solución del problema de optimización inherente al cálculo de las componentes principales disjuntas.

2.2 La metodología del Análisis de Componentes Principales Disjuntas

DPCA se fundamenta en expresar a la matriz $\mathbf{X}$ de tamaño $I \times J$ como el producto:

$$\mathbf{X} = \mathbf{AB}^T \tag{2.1.1}$$

Donde:

$\mathbf{A} = \lceil \lfloor a_{ij}$ matriz de tamaño $I \times Q$ de las coordenadas de los individuos en el espacio $\rfloor \rceil$

reducido Q dimensional de las componentes disjuntas.

$\mathbf{B} = \left[b_{ij} \right]$ matriz de tamaño $J \times Q$ de las coordenadas de las J variables en el espacio reducido Q dimensional de las componentes disjuntas, y está sujeta a las siguientes restricciones:

1 $\sum_{j=1}^{J} b_j^2 = 1, \quad q = 1,\ldots,Q$

2. $\sum_{j=1}^{J} \left(b_{jq} b_{jr} \right)^2 = 0 \quad q = 1,\ldots,Q-1; \quad r = q+1,\ldots,Q$

3 $\sum_{q=1}^{Q} b_{ij}^2 > 0, \quad j = 1,\ldots,J$

La primera restricción nos indica que cada columna es de norma unitaria y por ende no puede estar llena de ceros. La segunda restricción nos dice que dos columnas diferentes son ortogonales. Y la tercera, establece que ninguna fila está llena de ceros.

Como se trata de una aproximación a bajo rango, tenemos: $\mathbf{X} = \mathbf{A}\mathbf{B}^T + \mathbf{E}$, (2.1.2)

donde **E** es el error de aproximación. De (2.1.2), tenemos:

$$\mathbf{E} = \mathbf{X} - \mathbf{A}\mathbf{B}^T \tag{2.1.3}$$

Minimizar la norma de Frobenius al cuadrado del error **E** es equivalente a minimizar la función objetivo $F(\mathbf{A},\mathbf{B}) = \left\| \mathbf{X} - \mathbf{A}\mathbf{B}^T \right\|^2$ que corresponde a la suma de cuadrados residuales. Se tiene el siguiente problema de minimización:

$$\begin{cases} \min F(\mathbf{A},\mathbf{B}) = \left\| \mathbf{X} - \mathbf{A}\mathbf{B}^T \right\|^2 \\ \text{s.t } \mathbf{A},\mathbf{B} \text{ como se describe arriba} \end{cases} \tag{2.1.4}$$

La matriz de datos **X** es una matriz conocida, por tanto, a lo largo del análisis $\|\mathbf{X}\|$ es constante, minimizar $\dfrac{\left\| \mathbf{X} - \mathbf{A}\mathbf{B}^T \right\|^2}{\|\mathbf{X}\|^2}$ es equivalente a minimizar $\left\| \mathbf{X} - \mathbf{A}\mathbf{B}^T \right\|^2$. Razón por la

cual vamos a utilizar $F(\mathbf{A},\mathbf{B}) = \frac{\left\|\mathbf{X}-\mathbf{AB}^T\right\|^2}{\left\|\mathbf{X}\right\|^2}$, el error relativo, como función objetivo. F mide el grado de ajuste (fitness) de la aproximación a bajo rango.

A continuación, se presenta un algoritmo para minimizar F cuyo propósito es el de determinar Q componentes principales que generen un subespacio Q-dimensional, en el cual proyectar los datos originales, con la particularidad de que cada variable original contribuya a una sola componente y nada más que a una. Cabe recordar que estas componentes al ser disjuntas son ortogonales.

Para alcanzar el mínimo global de la función objetivo se implementa el algoritmo descrito a continuación.

2.2.1 Algoritmo DPCA

Este algoritmo es una adaptación del método Disjoint HJ Biplot al cálculo de componentes principales disjuntas, es un algoritmo iterativo, para el que se necesita una inicialización o paso cero**:**

Paso cero

Se dispone de una matriz de datos $\mathbf{X}$ de tamaño $I \times J$. Se elige Q el número de componentes disjuntas a considerar.

Se genera la matriz binaria estocástica por filas $\mathbf{V}_0$, de tamaño $J \times Q$, en forma aleatoria, tal que:

$$v_{jq} = \begin{cases} 1 \text{ si la variable } j \text{ contribuye a la componente } q \\ 0 \qquad\qquad\qquad \text{de otra forma} \end{cases} \tag{2.1.5}$$

que satisfaga las restricciones:

$$\sum_{q=1}^{Q} v_{jq} = 1, \quad j = 1,\ldots,J \tag{2.1.6}$$

$$\sum_{j=1}^{J} v_{jq} > 0, \quad q = 1, \ldots, Q \tag{2.1.7}$$

(2.1.6) asegura que exista un “uno” y nada más que uno en cada fila.

(2.1.7) verifica que no existan columnas llenas de “ceros”

De (2.1.6) y (2.1.7) se tiene:

$$\sum_{j=1}^{J} v_{jq} v_{jr} = 0, \quad q \neq r \tag{2.1.8}$$

Lo que significa que las columnas son ortogonales.

Así, por ejemplo, si J=5 y Q=3, $\mathbf{V}_0$ podría ser:

$$\mathbf{V}_0 = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$\mathbf{V}_0$ es una matriz de particiones, puesto que indica de qué forma van a estar repartidas las variables originales en las componentes disjuntas seleccionadas.

Se debe asegurar que ninguna componente tenga todas las cargas nulas. Si así sucede, la componente que tenga mayor número de variables asignadas se divide aleatoriamente en dos. Una de las mitades se pasa a la componente que tiene la columna de “ceros”.

2.2.2 Construcción de la matriz de cargas B:

Con $\mathbf{X}$ y $\mathbf{V}_0$ se calcula la matriz de cargas $\mathbf{B}_0$, como se explica a continuación:

Para cada componente q=1,...,Q, se genera una submatriz $\mathbf{W}_{0q}$ de la matriz de datos $\mathbf{X}$:

$\mathbf{W}_{0q}$ tiene I filas, al igual que $\mathbf{X}$ y tiene por columnas las correspondientes a las variables indicadas por los "unos" en la columna q de $\mathbf{V}_0$.

En el ejemplo mencionado:

$$\mathbf{W}_{01} = \begin{pmatrix} X_1 & X_2 \\ x_{11} & x_{12} \\ x_{21} & x_{22} \\ x_{31} & x_{32} \\ \vdots & \vdots \\ x_{I1} & x_{I2} \end{pmatrix}, \quad \mathbf{W}_{02} = \begin{pmatrix} X_4 \\ x_{14} \\ x_{24} \\ x_{34} \\ \vdots \\ x_{I4} \end{pmatrix}, \quad \mathbf{W}_{03} = \begin{pmatrix} X_3 & X_5 \\ x_{13} & x_{15} \\ x_{23} & x_{25} \\ x_{33} & x_{35} \\ \vdots & \vdots \\ x_{I3} & x_{I5} \end{pmatrix}$$

Cada $\mathbf{W}_{0q}$ se expresa mediante su SVD de la forma:

$$\mathbf{W}_{0q} = \tilde{\mathbf{R}}\Lambda\mathbf{T}^T \tag{2.1.9}$$

En total se tienen Q SVD's de la forma (2.1.9). Se construye la matriz $\mathbf{B}_0$ de la siguiente forma: La columna q de $\mathbf{B}_0$ es el vector singular derecho correspondiente al mayor valor singular de la q-ésima SVD. Esta columna contendrá las cargas correspondientes a las variables consideradas en la q-ésima columna de $\mathbf{V}_0$, con ceros en las posiciones que no están consideradas en dicha columna q. Puesto que se tienen Q SVD's el número de columnas de $\mathbf{B}_0$ es Q.

Con la matriz $\mathbf{B}_0$, se obtienen las coordenadas para los objetos:

$$\mathbf{A}_0 = \mathbf{X}\mathbf{B}_0 \tag{2.1.10}$$

Por último, se calcula la función objetivo $F_0(\mathbf{A}_0, \mathbf{B}_0)$

Terminado el paso inicial, se inicia el proceso iterativo.

Paso *k*:

Se supone que ya se han ejecutado *k*-1 pasos del algoritmo. Se dispone de las matrices $\mathbf{V}_{k-1}, \mathbf{A}_{k-1}, \mathbf{B}_{k-1}$ y de F_{k-1}.

Comenzamos actualizando la matriz $\mathbf{V}_k$:

Sea la *j*-ésima fila, con *j*=1,...,*J*. En la *q*-ésima posición de esta fila (*q*=1,..., *Q*) se coloca un "uno" y "ceros" en el resto de la fila. Manteniendo sin cambio el resto de filas de la matriz $\mathbf{V}_{k-1}$, a esta matriz se la nota por $\mathbf{V}_{kq}$, con esta nueva matriz de particiones obtenemos las matrices $\mathbf{W}_{kq}$ y las respectivas matrices $\mathbf{A}_{kq}$ y $\mathbf{B}_{kq}$ y se evalúa la función objetivo $F_{kq}(\mathbf{A}_{kq}, \mathbf{B}_{kq})$. A la posición en la que $F_{kq}(\mathbf{A}_{kq}, \mathbf{B}_{kq})$ sea máxima se le asigna el "uno" de dicha fila. Se continua con este proceso de reubicación de los "unos" en todas las filas de $\mathbf{V}_{k-1}$. A la matriz resultante se la denomina $\mathbf{V}_k$. Puesto que en cada fila la posición que contiene al "uno" recorre todas las *Q* entradas de la fila *j*, y que hay en total *J* filas, el número de descomposiciones en valores singulares para determinar $\mathbf{V}_k$ es *J*x*Q*.

Si en el proceso se genera alguna columna de $\mathbf{V}_k$ con todos los elementos iguales a cero, se actúa de igual manera a la que se explicó en la construcción de $\mathbf{V}_0$.

Con la matriz de particiones $\mathbf{V}_k$ se obtienen las correspondientes matrices $\mathbf{W}_{kq}$ y a partir de éstas la matriz de cargas $\mathbf{B}_k$, de la manera ya indicada, luego se evalúa la matriz de scores $\mathbf{A}_k$:

$$\mathbf{A}_k = \mathbf{X}\mathbf{B}_k \tag{2.1.11}$$

Por último, se calcula $F_k(\mathbf{A}_k, \mathbf{B}_k)$

Se verifica si se satisface el criterio de parada.

Condición de parada:

Se fija un valor de tolerancia $\varepsilon > 0$. Si $\left|F_k - F_{k-1}\right| < \varepsilon$ se detiene el procedimiento.

Si es así, se termina el proceso iterativo y se considera como solución la obtenida en este paso. Si no se satisfacen las condiciones de parada, se vuelve a iterar hasta que se cumplan las condiciones de parada.

El siguiente algoritmo sintetiza los pasos necesarios para llevar a cabo el procedimiento de cálculo para realizar el DPCA, que lo llamamos clásico para diferenciarlo del algoritmo que utiliza PSO descrito más adelante:

Algoritmo DPCA clásico

1: Leer $\mathbf{X}$, Q, nIter (número de iteraciones)

2: Inicializar $\mathbf{V}_0$, $k == 0$

3: Inicio de iteraciones

3.1: Sea $k == k + 1$

3.2: Actualizar $\mathbf{V}_k$

3.3: Calcular la matriz de particiones $\mathbf{W}_{kq}$ para $q == 1,.,Q$

3.4: Calcular la SVD de $\mathbf{W}_{kq}$ para $q = 1,.,Q$

3.5: Obtener $\mathbf{B}_k$, $\mathbf{A}_k$

3.6: Actualizar $F_k == F(\mathbf{A}_k, \mathbf{B}_k)$

4: Repita las iteraciones hasta que k =nIter

5: Mostrar los resultados $\mathbf{A}_k$ y $\mathbf{B}_k$

La manera en que se construyen las matrices $\mathbf{V}_k$ no es flexible, puesto que a medida que se desciende por filas en la construcción de esta matriz las posiciones que contienen al "uno" de cada fila van quedando fijas y no se alteran. Para evidenciar esta situación recurrimos al siguiente ejemplo:

Se han determinado los "unos" de las tres primeras filas, estas posiciones son inamovibles y se está determinando la posición del "uno" de la penúltima fila:

$$\begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ \boxed{?} & \boxed{?} & \boxed{?} \\ 0 & 0 & 1 \end{pmatrix}$$

si resulta que

$$\begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ \boxed{0} & \boxed{0} & \boxed{1} \\ 0 & 0 & 1 \end{pmatrix}$$

se pasa a determinar el "uno" de la última fila.

Resultando la matriz $\mathbf{V}_k$ igual a

$$\begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ \boxed{1} & 0 & 0 \end{pmatrix}.$$

Este procedimiento impide probar con matrices de la forma

$$\begin{pmatrix} \boxed{1} & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \quad \text{o}$$

$$\begin{pmatrix} \boxed{1} & 0 & 0 \\ 0 & 0 & \boxed{1} \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$$

pues los "unos" de las primeras filas están fijos, perdiéndose posibles particiones que mejoren la función objetivo.

Otra opción sería tomar en cuenta a todas las posibles matrices binarias $\mathbf{V}_k$, pero el número de posibles particiones es extraordinariamente alto, aun para valores pequeños de J y Q tal como se muestra en la siguiente sección, tabla 2.1.

2.3 Metodología de la Optimización por Enjambre de Partículas.

El método de optimización denominado "por enjambre de partículas" (Particle Swarm Optimization) (Kennedy & Eberhart, 1995), (Poli et al., 2007) se inspira en la conducta de sociedades biológicas (abejas, peces, aves) que comparten comportamientos individuales y sociales:

- Las partículas o individuos son atraídos hacia la comida
- En cualquier instante los individuos conocen su cercanía a la comida. La cercanía es estimada a través del denominado fitness y es el valor que asigna la función objetivo a la posición donde se encuentra la partícula.

- Cada partícula o individuo recuerda su posición más cercana a la comida. Es el conocimiento histórico individual.
- Las partículas comparten información sobre cuál ha sido su posición más cercana a la comida con las partículas más próximas a ella. Es el conocimiento histórico de su vecindario.

Mediante dos reglas de interacción los individuos adaptan su comportamiento al de los individuos con más éxito de su entorno (Imran et al., 2013).

El objetivo del método PSO es que las partículas exploren el conjunto de soluciones factibles en busca del óptimo. La población inicial de partículas se mantiene constante a través del proceso de búsqueda (Eberhart & Yuhui, 2001).

En cada momento *t* a cada partícula *i* le corresponde una posición $\mathbf{x}_i^t$, una velocidad $\mathbf{v}_i^t$ y un valor de fitness $F(\mathbf{x}_i^t)$ con $\mathbf{x}_i^t, \mathbf{v}^t \in \mathbb{R}^n$.

El algoritmo PSO es iterativo y en cada instante del tiempo cada partícula "sabe":

- Cuál ha sido su mejor posición respecto al óptimo (personal best, p-best $\mathbf{p}_i$). Es la posición donde alcanzó su mejor fitness.
- Cuál ha sido la mejor posición de sus vecinos (global best, g-best $\mathbf{p}_g$). Es la posición donde su vecindario alcanzó el mejor fitness.

Las partículas actualizan su posición de acuerdo a las ecuaciones:

$$\mathbf{x}_i^{t+1} = \mathbf{x}^t + \mathbf{v}^{t+1}, \text{ donde} \tag{2.2.1}$$

$$\mathbf{v}_i^{t+1} = \underbrace{\phi_0 \mathbf{v}_i^t}_{INERCIA} + \underbrace{\phi_{1,1,t} r (\mathbf{p}_i - \mathbf{x}_i^t)}_{INFORMACION\ LOCAL} + \underbrace{\phi_{2,2,t} r (\mathbf{p}_g - \mathbf{x}_i^t)}_{INFORMACION\ GLOBAL} \tag{2.2.2}$$

En estas ecuaciones se considera la siguiente información:

$\mathbf{x}_i^t$: Posición de la partícula *i* al tiempo *t*

$\mathbf{v}_i^t$: Velocidad de la partícula *i* al tiempo *t*, es el gradiente, o dirección en la cual se moverá la partícula:

$$\mathbf{v}^t = \mathbf{x}_i^t - \mathbf{x}^{t-1} \tag{2.2.3}$$

La inercia acelera o ralentiza a la partícula. Si la partícula se mueve muy rápido recorrerá en pocos pasos el conjunto de búsqueda *S* y eventualmente puede omitir regiones donde se encuentre el óptimo buscado (Poli et al., 2007). Si la partícula se mueve lentamente, la búsqueda será más minuciosa, pero a cambio el tiempo de cálculo puede incrementarse de forma notable. ϕ_0 es el coeficiente de inercia y determina si el movimiento es lento o rápido. Para que no se den los extremos de movimientos muy rápidos o muy lentos es necesario fijar un mínimo y un máximo para este parámetro (Eberhart & Yuhui, 2001).

Información Local o Componente Cognoscitivo: es la atracción a la posición de esa partícula que obtuvo el mejor ajuste (mejor personal) y depende de la distancia existente entre la posición de la mejor solución individual, hasta el instante *t,* $\mathbf{p}_i$, y la posición actual al tiempo *t* , $\mathbf{x}_i^t$. ϕ_1 es la tasa de aprendizaje individual.

Información Global o Componente Social: es la atracción a la posición de la partícula que haya obtenido el mejor ajuste a nivel general (mejor global) y depende de la distancia existente entre la posición de la mejor solución del vecindario, hasta el instante *t,* $\mathbf{p}_g$, y la posición actual al tiempo *t* , $\mathbf{x}_i^t$. ϕ_2 es la tasa de aprendizaje social.

$r_{1,t}, r_{2,t}$ son números aleatorios entre cero y uno, representan perturbaciones locales y globales respectivamente.

Se puede interpretar a $\mathbf{v}_i^t$ como la innovación de la partícula *i* al tiempo *t*.

2.3.1 Algoritmo PSO genérico:

De manera general el funcionamiento del algoritmo PSO se puede describir como un proceso iterativo y estocástico que se puede detallar de la siguiente manera:

Inicio (algoritmo)

Inicializar la posición y la velocidad de las partículas aleatoriamente.

Mientras no se alcance la condición de parada y la partícula permanezca en el conjunto de soluciones factibles **hacer**

1: Para cada partícula $p = 1,..,P$ hacer:

1.1 : Actualizar su velocidad considerando (2.2.2)

1.2: Actualizar la nueva posición de acuerdo a (2.2.1)

1.3: Calcular el fitness de la partícula en la nueva posición

1.4: Actualizar su mejor personal

2: Actualizar el mejor global

3: Devolver el mejor global

Fin (algoritmo)

Como criterio de parada se considera el número de iteraciones dado por el usuario. No se va a utilizar el procedimiento de fijar un $\varepsilon > 0$ y recurrir a la desigualdad $|F_k - F_{k-1}| < \varepsilon$ como condición de parada, ya que en un algoritmo evolutivo puede suceder que en dos iteraciones sucesivas no haya cambio de la función objetivo.

En el algoritmo las partículas son matrices binarias de la forma **V** indicada en la sección 2.2. En cada iteración, al actualizar la posición, la partícula abandona el conjunto de soluciones factibles, puesto que si bien la matriz de posiciones es binaria la matriz de velocidades no lo es.

Veamos un pequeño ejemplo, si en la etapa *k* tenemos una posición y una velocidad dadas respectivamente por:

$$\begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \text{ y}$$

$$\begin{pmatrix} -0.51 & 0.75 & 0.12 \\ -0.33 & 0.24 & 0.68 \\ -0.83 & -0.45 & 0.53 \\ 0.71 & 0.58 & 0.91 \\ 0.46 & -0.79 & 0.64 \end{pmatrix}$$

Las matrices de velocidad siempre tienen sus componentes en el intervalo $[-1, 1]$, como se explica en la sección 2.3.2. Al actualizar la posición en la etapa k+1 tenemos una nueva posición que claramente no es una matriz binaria que satisface las restricciones requeridas:

$$\begin{pmatrix} 0.49 & 0.75 & 0.12 \\ -0.33 & 0.24 & 0.68 \\ -0.83 & 0.55 & 0.53 \\ 0.71 & 0.58 & 0.09 \\ 0.46 & -0.79 & 0.64 \end{pmatrix}$$

La solución propuesta innovadora a este problema es la siguiente:

1. En cada fila se elige la entrada de mayor valor absoluto. Estas posiciones serán ocupadas por “unos” el resto por “ceros”. Este procedimiento, a diferencia de elegir los “unos” aleatoriamente, permite conservar la memoria de la partícula.
2. En el caso en que una columna resulte sólo con ceros, se elige la columna con mayor cantidad de “unos” localizamos el “uno” que provino del menor valor

absoluto, y lo cambiamos a la columna llena de ceros. La matriz así obtenida satisface las condiciones del conjunto de soluciones factibles.

3. Si existieran dos columnas llenas de ceros, aplicamos el mismo procedimiento: elegimos los dos "unos" que provengan de los dos valores absolutos más pequeños y los pasamos a las columnas de ceros, un "uno" a cada columna.

Para ejemplificar el procedimiento anterior vamos a suponer que luego de la actualización obtenemos la siguiente matriz de posición:

$$\begin{pmatrix} 0.49 & \boxed{0.75} & 0.12 \\ -0.33 & 0.24 & \boxed{0.68} \\ \boxed{-0.83} & 0.55 & 0.53 \\ \boxed{0.71} & 0.58 & 0.09 \\ 0.46 & -0.79 & \boxed{1.64} \end{pmatrix}$$

Las posiciones dentro de los recuadros serán ocupadas por "unos" el resto por "ceros":

$$\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Si resulta una columna sólo con ceros, como en el siguiente caso:

$$\overbrace{\begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}}^{Posición} + \overbrace{\begin{pmatrix} -0.51 & 0.75 & 0.12 \\ 0.33 & 0.24 & 0.68 \\ -0.83 & -0.45 & 0.53 \\ 0.71 & -0.58 & -0.36 \\ 0.51 & -0.79 & -0.44 \end{pmatrix}}^{Velocidad} = \overbrace{\begin{pmatrix} 0.49 & \boxed{0.75} & 0.12 \\ \boxed{1.33} & 0.24 & 0.68 \\ \boxed{-0.83} & -0.45 & 0.64 \\ \boxed{0.71} & 0.42 & -0.36 \\ 0.51 & \boxed{-0.79} & 0.56 \end{pmatrix}}^{Nueva\ posición}$$

Obtendríamos una matriz binaria con una columna de ceros:

$$\begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ \boxed{1} & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$$

Para paliar este problema elegimos la primera columna, que es la que tiene mayor cantidad de “unos”, ubicamos el “uno” que provino del menor valor absoluto, en el ejemplo, el “uno” de la cuarta fila. A este “uno” lo cambiamos a la tercera columna y obtenemos la matriz binaria:

$$\begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & \boxed{1} \\ 0 & 1 & 0 \end{pmatrix}$$

Dicha matriz satisface las condiciones del conjunto de soluciones factibles.

Por último, que tal si hay un empate en los mayores valores absolutos de una fila. Por ejemplo:

$$\begin{pmatrix} 0.49 & \boxed{0.75} & \boxed{0.75} \\ -0.33 & 0.24 & 0.68 \\ -0.83 & 0.55 & 0.53 \\ 0.71 & 0.58 & 0.09 \\ 0.46 & -0.79 & 0.64 \end{pmatrix}$$

En esta situación se elige siempre el primero de ellos de izquierda a derecha. Dado que las entradas son números reales considerados como variable tipo “double", su representación va a tener un rango de entre -5x10-324 hasta 1.7x10308, por lo que la situación de dos entradas con valores absolutos iguales es muy poco probable.

2.3.2 Método Constrained Binary PSO Disjoint Components (CBPSO DC)

Sea **X** una matriz de datos con I objetos y J variables. El objetivo consiste en realizar una reducción de dimensiones para poder trabajar con Q componentes ($Q < J$) con la característica de que sean disjuntos (cada variable debe estar en una componente y sólo una, y todas las componentes deben tener al menos una variable).

Para ello, en el método Disjoint PCA clásico se utiliza una codificación binaria en una matriz **V** que permite posteriormente la construcción de la matriz **B** de cargas en base a descomposiciones SVD como se detalla en (Ferrara et al., 2016), (Macedo & Freitas, 2015)(Macedo & Freitas, 2015). Ambas matrices **V** y **B** son de tamaño JxQ.

La matriz $\mathbf{V}$ es una matriz binaria de entradas:

$$v_{jq} \in \{0,1\}$$

donde v_{jq} es una entrada cualquiera de la matriz $\mathbf{V}$ que satisfacen las restricciones dadas en (2.1.6), y (2.1.7)

Sea S el conjunto de soluciones factibles, es decir, el conjunto de todas las matrices binarias $\mathbf{V}$ que satisfacen las restricciones mencionadas. Se plantea el siguiente modelo de optimización:

$$\begin{cases} \min F(\mathbf{V}) = \| \mathbf{X} - \mathbf{A}\mathbf{B}^T \|^2 \\ \mathbf{B} = T(\mathbf{V}) \text{ and } \mathbf{A} = \mathbf{X}\mathbf{B} \\ \text{s.t } \mathbf{V} \in S \end{cases} \tag{2.2.4}$$

Donde T es un operador matricial que permite transformar la matriz binaria **V** en la matriz de cargas **B** de la forma descrita en "*Construcción de la matriz de cargas B*" en la sección

2.2.2. Es importante recalcar que para realizar el cálculo $\mathbf{B} = T(\mathbf{V})$ el operador T realiza tantas descomposiciones SVD como componentes disjuntos Q se deseen tener.

En el problema de optimización anterior se busca dentro del conjunto S aquella matriz binaria $\mathbf{V}$ * que minimiza la función objetivo F (esto es $\forall\, \mathbf{V} \in S\ F(\mathbf{V}^*) \leq F(\mathbf{V})$). En otras palabras, se desea aproximar la matriz de datos $\mathbf{X}$ lo mejor posible. Claramente se trata de un problema de optimización binario con restricciones.

Para ilustrar la complejidad del problema, supongamos que el espacio de dimensión reducida deseado emplea únicamente dos componentes disjuntos. Con J variables, el conjunto S tiene $2^J - 2$ soluciones factibles. Con tres componentes disjuntos y J variables, S tiene $3^J + 3 - 3 \cdot 2^J$ soluciones factibles. En la tabla 3.1 se muestra la cardinalidad se S para diferentes valores de J y Q:

	Q=2	Q=3
J=10	1022	55980
J=15	32766	14250606
J=20	1048574	3483638676
J=30	10737418	2.058879x10^14

Tabla 2.1: Crecimiento exponencial de soluciones factibles

Se trata de un problema de una alta complejidad computacional (NP-Hard) debido a la explosión combinatoria presente. En este trabajo se muestra el diseño de un algoritmo PSO binario con restricciones para resolver el problema de optimización combinatoria.

Algoritmo CBPSO DC

El primer paso del algoritmo es inicializar las partículas, lo que se traduce en que ellas se ubiquen en una posición inicial. El número de partículas P es un parámetro de entrada

importante en esta etapa. La posición inicial de cualquier partícula es una solución factible del problema, es decir, un elemento del conjunto S.

La función fitness está dada por:

$$\frac{\left\|\mathbf{X}-\mathbf{A}\mathbf{B}^{T}\right\|^{2}}{\left\|\mathbf{X}\right\|^{2}} \tag{2.2.5}$$

y viene a representar el error cuadrático relativo en forma de tanto por uno. Como ya se mencionó en la sección 2, debido a que la norma de **X** es una constante a lo largo de la ejecución del programa minimizar (2.2.5) equivale a minimizar $F(\mathbf{V})=\left\|\mathbf{X}-\mathbf{A}\mathbf{B}^{T}\right\|^{2}$. Al ser una medida del error de aproximación se tiene que a menor valor de la función fitness mejor solución.

En el algoritmo PSO existen dos tipos de variables, las de carácter individual y las colectivas o globales. En las individuales, cada partícula almacena sus propios valores. En las globales, los valores son compartidos por todas las partículas.

Variables Individuales

Para una partícula p en particular se almacena en memoria la siguiente información:

V_p representa la matriz binaria $\mathbf{V}$ para la partícula p en la que se encuentra (posición actual binaria).

$B_p = T(V_p)$ es la matriz de cargas $\mathbf{B}$ para la partícula p en la que se encuentra (posición actual real).

$Fit_p = F(V_p)$ corresponde al valor de la función objetivo evaluada en la posición actual de la partícula p.

$bestV_p$ es la mejor matriz binaria $\mathbf{V}$ encontrada por la partícula p hasta el momento.

$bestB_p = T(bestV)$ representa la mejor matriz de cargas $\mathbf{B}$ encontrada por la partícula p hasta el momento (personal best).

$bestFit_p = F(bestV_p)$ es el valor de la función objetivo evaluada en la mejor solución encontrada por la partícula p hasta el momento.

Vel_p Velocidad actual de la partícula p. Se usa para generar una perturbación sobre la posición actual de la partícula con la intención de ubicarla en una nueva posición. Por ende Vel_p es una matriz de tamaño JxQ, y como parte del diseño del algoritmo, cada entrada de esta matriz se encuentra en el intervalo $[-1,1]$

Variables Globales

A nivel de todo el enjambre se recoge la siguiente información:

$bestV$: es la mejor matriz binaria $\mathbf{V}$ encontrada por el cúmulo de partículas hasta el momento (global best).

$bestB = T(bestV)$ es la mejor matriz de cargas $\mathbf{B}$ encontrada por el cúmulo de partículas hasta el momento.

$bestFit = F(bestV)$ representa el valor de la función objetivo evaluada en la mejor solución encontrada por el cúmulo de partículas hasta el momento

Paso 1: Inicialización

Durante la inicialización de las partículas se realizan las siguientes tareas:

1: Para cada partícula p, desde $p = 1$ hasta $p = P$

1.1 : Se genera aleatoriamente la matriz V_p

1.2 : Se calcula $B_p = T(V)$

1.3 : Se calcula $Fit_p = F(V_p)$

1.4 : $bestV_p = V_p$

1.5 : $bestB_p = B_p$

1.6 : $bestFit_p = Fit_p$

1.7 : Se genera aleatoriamente Vel_p (velocidad inicial)

2: De entre todas las P partículas se busca la partícula con la mejor posición inicial binaria, que corresponde a la que tiene mejor fitness. Sea $p*$ la partícula con la mejor posición inicial. Entonces:

2.1 : $bestV = V_{p*}$

2.2 : $bestB = B_{p*}$

2.3 : $bestFit = bestFit_{p*}$

Paso 2: Proceso iterativo

El segundo paso del algoritmo es el más importante porque corresponde a las iteraciones:

En una iteración cualquiera todas las partículas deben moverse a una nueva posición. Cada posición es una solución factible para el problema, es decir, no le es permitido a una partícula ubicarse en la posición de una solución no factible. En esta etapa se tienen los siguientes parámetros:

nIter: número de veces que debe iterar el algoritmo.

minIner: valor de inercia mínimo. Corresponde al valor de la inercia en la última iteración.

maxIner: valor de inercia máximo. Corresponde al valor de la inercia en la primera iteración.

wCognition: Peso cognitivo, sirve para controlar el componente cognitivo de la nueva velocidad.

wSocial: Peso Social, sirve para controlar el componente social de la nueva velocidad.

Para determinar la nueva posición de una partícula p se necesita calcular una nueva velocidad que depende de tres componentes: el primero de ellos es la velocidad en la iteración anterior que se controla con la inercia (esta inercia es en sentido físico, nada tiene que ver con el concepto de inercia que se utiliza en el análisis multivariante en el que denota la variabilidad de un conjunto de datos). El segundo componente es el cognitivo que lo manejamos con el peso cognitivo y el tercer componente es el social que lo controlamos con el peso social. La nueva velocidad sumada a la posición actual de la partícula permite obtener así una nueva posición. Es muy importante vigilar que la nueva posición corresponda a una solución factible del problema, para ello se define un operador matricial que nos asegure que la matriz-partícula no abandone el conjunto de soluciones factibles, como se detallará más adelante.

El valor de la inercia va disminuyendo linealmente de iteración en iteración. Iniciamos las iteraciones con un valor máximo de inercia y finalizamos las iteraciones con un valor mínimo de inercia. Entonces, para una iteración $k\left(1 \le k \le nIter\right)$ el valor de la inercia está dado por la ecuación:

$$Inercia = \text{maxInercia} - \left(\frac{\text{maxInercia} - \text{minInercia}}{\text{nIter}}\right)k \qquad (2.2.6)$$

La ecuación (2.2.6) permite que, conforme se itera, la nueva velocidad vaya dependiendo menos de la velocidad anterior, y los componentes cognitivo y social la afecten de mayor manera. El valor de la nueva velocidad para una partícula p en la iteración k se obtiene mediante la siguiente ecuación:

$$newVel_p = Inercia * Vel_p + \rho_1 * wCognition * \left(bestB_p - B_p\right) + \rho_2 * wSocial * \left(bestB - B_p\right)$$

(2.2.7)

Los valores $\rho_1, \rho_2 \in [0,1]$ son aleatorios con el fin de que la nueva velocidad de la partícula p tenga un componente estocástico.

Es de observar que $newVel_p$ es una matriz de tamaño JxQ que en general no cumple con la restricción de que sus entradas se encuentren en el intervalo $[-1,1]$ por lo que se debe aplicar un operador L tal que la matriz $Vel_p = L(newVel)$ tenga todas sus entradas en el intervalo $[-1,1]$.

Sea $z \in \mathbb{R}$ una entrada cualquiera de la matriz $newVel_p$, se define el operador L por la combinación lineal convexa:

.

$$L(z) = 2\text{sigmo}(z) - 1 \tag{2.2.8}$$

Donde:

$$\text{sigmo}(z) = \frac{1}{1+e^{-z}} \in [0,1] \tag{2.2.9}$$

Por tanto $L(z) \in [-1,1]$.

Se aplica el operador L a todas las entradas de la matriz $newVel_p$ tal como se explicó anteriormente.

Luego, se calcula una nueva posición real temporal mediante la ecuación:

$$tempB_p = B_p + Vel_p \tag{2.2.10}$$

A la matriz $tempB_p$ se le aplica nuevamente el operador L para que sus entradas se encuentren en el intervalo $[-1,1]$. Ahora, mediante la aplicación del operador matricial E a la matriz $tempB_p$, descrito más abajo, se obtiene la nueva posición binaria y real de la partícula p en el conjunto de soluciones factibles:

$$V_p = E\left(tempB_p\right) \tag{2.2.11}$$

$$B_p = T\left(V_p\right) \tag{2.2.12}$$

Una pieza clave de la implementación de este algoritmo PSO binario es el operador E que convierte a la matriz $tempB_p$ en la matriz V_p que pertenece al conjunto de soluciones factibles y se define de la siguiente manera:

a. Para cada fila $j = 1,2,\cdots,$ de la matriz $tempB_p$ se identifica aquella columna q (con $q = 1,\ldots,Q$) donde se encuentra el mayor elemento en valor absoluto, y de esta manera, se ubica un uno en la posición (j,q) de la matriz V_p. Los demás elementos de la fila j son fijados a cero. Esto debe hacerse por cada fila. Si el mayor elemento en valor absoluto se encuentra en dos o más columnas de la misma fila, se toma la posición de la primera columna de izquierda a derecha para hacer la asignación del uno correspondiente.

b. Si al concluir el procedimiento (a) existe una columna q en la matriz V_p en la que todos sus elementos son cero, se debe identificar en esta misma matriz, aquella columna q' que tenga la mayor cantidad de números uno. Luego, en la columna q' de la matriz $tempB_p$ se debe identificar la fila j

donde se encuentre el valor más bajo en valor absoluto. Finalmente, en la matriz V_p se asigna uno en la posición (j,q) y cero en la posición (j,q'). Se procede de la misma forma si existen dos o más columnas en la matriz V_p en las que todos sus elementos son cero.

De esta manera se cumplen las restricciones para las matrices binarias $\mathbf{V}$ que pertenecen al conjunto de soluciones factibles S.

Fase de actualización:

La actualización de las partículas, en cada iteración, consiste en las siguientes etapas:

1: Para cada iteración k, desde $k = 1$ hasta $k = nIter$

1.1: Se calcula $Inercia$

1.2: Para cada partícula p, desde $p = 1$ hasta $p = P$

1.2.1: Se calcula $newVel_p$

1.2.2: Se calcula $Vel_p = L(newVel)$

1.2.3: Se calcula $tempB_p = B_p + Vel_p$

1.2.4: Se calcula $V_p = E(tempB_p)$

1.2.5: Se calcula $B_p = T(V)$

1.2.6: $Fit_p = F(V_p)$

1.2.7: Si $Fit_p < bestFit_p$

1.2.7.1 : $bestV_p = V_p$

1.2.7.2 : $bestB_p = B_p$

1.2.7.3 : $bestFit_p = Fit_p$

1.2.8 : Si $Fit_p < bestFit$

1.2.8.1 : $bestV = V_p$

1.2.8.2 : $bestB = B_p$

1.2.8.3 : $bestFit = Fit_p$

2: Mostrar la mejor solución encontrada.

2.3.3 Propuesta de cálculo de la varianza explicada por los ejes disjuntos

Como se puede advertir el método CBPSO no calcula la matriz diagonal Λ, que contiene a los valores propios, razón por la que se desconocen los valores singulares de $\mathbf{X}$ y por ende los valores propios de $\mathbf{X}^t\mathbf{X}$. En este trabajo se propone estimar los valores propios correspondientes a cada una de las componentes disjuntas por medio del cálculo de la varianza que exhiben los individuos en el nuevo sistema referencial. Esto habilita encontrar un porcentaje de explicación de la varianza para cada uno de los ejes del sistema referencial disjunto y los respectivos planos principales. Para más detalle veamos:

Sea $\mathbf{X}$ una matriz de datos, centrada y de tamaño *I*x*J* y sea $\text{var}(\mathbf{X}) = \mathbf{S}$ su matriz de varianzas-covarianzas, se define la variación total de **X** por

$$tr(\mathbf{S}) = \sigma^2_{J} + \sigma^2 + \cdots + \sigma^2_J = \sum_{j=1}^{J} \sigma_j^2 \qquad (2.2.13)$$

donde $\sigma_1^2, \sigma^2, \ldots, \sigma_J^2$ son las varianzas de las *J* variables contenidas en **X**, es decir, $\text{var}(X_j) = \sigma^2$. **A**lternativamente tenemos:

$$\text{tr}(\mathbf{S}) = \alpha_1 + \alpha_2 + \cdots + \alpha_J \tag{2.2.14}$$

donde $\alpha_1, \alpha_2, \cdots, \alpha_J$ son los valores propios de **S** (Cuadras, 2014)**.**

Sea **B** una matriz de rotación *JxJ*, cualquiera, es decir, **B** es ortogonal ($\mathbf{B}^T\mathbf{B} = \mathbf{I}$). Los vectores columna de **B** definen un nuevo sistema referencial rotado. Cada nuevo eje es una combinación lineal de las variables originales y como tal representa una nueva variable.

Sea **A** una transformación o rotación de **X** dada por $\mathbf{A} = \mathbf{XB}$, su matriz de varianzas-covarianzas está dada por:

$$\begin{aligned} \mathbf{\Sigma} &= \text{var}(\mathbf{A}) \\ &= \mathbf{B}^T\mathbf{SB} \end{aligned} \tag{2.2.15}$$

Puesto que las columnas de **A** contienen las coordenadas de los individuos respecto al nuevo sistema referencial, la varianza de cada columna representa la varianza de las nuevas variables obtenidas por la rotación **B** .

La variación total de **A** es

$$\begin{aligned} \text{tr}(\mathbf{\Sigma}) &= \text{tr}(\mathbf{B}^T\mathbf{SB}) \\ &= \text{tr}(\mathbf{SBB}^T) \\ &= \text{tr}(\mathbf{SI}) \\ &= \text{tr}(\mathbf{S}) \end{aligned} \tag{2.2.16}$$

Es decir, la variación total de **X** y **A** es la misma, la rotación **B** no afecta a la variación total.

Si $\beta_1, \beta_2, \cdots, \beta_J$ son los vectores propios de la matriz $\mathbf{\Sigma}$ ordenados decrecientemente, entonces:

La varianza de la j-ésima nueva variable notada por A_j es $\text{var}\left(A_j\right) = \beta$ con $j = 1,\ldots, J$

y

$$\text{tr}\left(\mathbf{\Sigma}\right) = \beta_1 + \beta_2 + \cdots + \beta_J \tag{2.2.17}$$

De (2.2.13), (2.2.14) y (2.2.17):

$$\begin{aligned} \sigma_1^2 + \sigma^2 + \cdots + \sigma_J^2 &= \alpha + \alpha + \cdots + \alpha_J \\ &= \beta_1 + \beta_2 + \cdots + \beta_J \end{aligned} \tag{2.2.18}$$

Por lo que se pueden utilizar los coeficientes β_i para determinar los porcentajes de explicación de los ejes disjuntos.

Así, el porcentaje de la variación total explicada por el q-ésimo eje disjunto es:

$$\frac{\beta_q}{\sum_{i=1} \beta_i} \times 100\% = \frac{\beta_q}{\sum_{j=1} \sigma^2} \times 100\% \tag{2.2.19}$$

El porcentaje de la variación total explicada por el plano generado por los ejes disjuntos q y r es:

$$\frac{\beta_q + \beta_r}{\sum_{i=1} \beta_i} \times 100\% = \frac{\beta_q + \beta_r}{\sum_{j=1} \sigma^2} \times 100\% \tag{2.2.20}$$

La cantidad $\sum_{j=1}^{J} \sigma_j^2$ se puede obtener directamente de la matriz $\mathbf{X}$ y los coeficientes β_j se puede estimar a partir de $\text{var}\left(A_j\right)$

En el caso Q=J, al aplicar el algoritmo CBPSO para obtener las componentes disjuntas, se obtiene la matriz de cargas $\mathbf{B} = \mathbf{I}_Q$ y por tanto $\mathbf{A} = \mathbf{XB} = \mathbf{X}$. Si $Q < J$, las columnas de $\mathbf{B}$ son disjuntas y de norma uno, por ende forman un conjunto ortonormal de Q vectores en $\mathbb{R}^J$, y además $\mathbf{B}'\mathbf{B} = \mathbf{I}_Q$.

De aquí que a la matriz $\mathbf{B}$ se la puede considerar como una matriz de proyección en el nuevo sistema referencial disjunto rotado y $\mathbf{A} = \mathbf{XB}$ es la matriz que contiene las coordenadas de los I objetos que se encuentran en dicho subespacio vectorial generado por las Q primeras componentes disjuntas.

Como se mencionó anteriormente, nuestra propuesta es utilizar la varianza de los scores, es decir, podemos aplicar (2.2.19) y (2.2.20). La varianza de las columnas de $\mathbf{A}$ nos sirve de indicador de la varianza explicada por cada eje disjunto obtenido, y para cada plano principal requerido.

2.4 Ejemplos Numéricos

2.4.1 Generador de matrices con estructura de componentes disjuntas

El criterio de parada utilizado en el algoritmo CBPSO DC es el número de iteraciones. Sin embargo, cada vez que el algoritmo encuentra una solución mejor (menor fit), la almacena junto con el tiempo de procesamiento que tomó encontrarla. La mejor solución encontrada por el algoritmo usualmente tiene un tiempo de procesamiento menor al tiempo que se necesita para que se ejecuten todas las iteraciones.

El objetivo en esta sección es construir un algoritmo de simulación que de forma aleatoria genere una matriz de datos, con una estructura ad hoc de fácil interpretación, y luego al aplicar una reducción dimensional por componentes principales disjuntas, el algoritmo CBPSO DC sea capaz de detectarla.

Supongamos que se tienen $x_1, \ldots, x_p$, p variables originales. Por otro lado $y_1, \ldots, y_q$ son las q variables latentes ($q < p$) con estructura disjunta. Considere la combinación lineal:

$$y_i = c_{1,i}x_1 + \ldots + c_{p,i}x_p \tag{2.3.1}$$

Si se desea que las m variables originales consecutivas x_j, x_{j+1}, ... , $x_{j+(m-1)}$ estén representadas en la variable latente y_i entonces los escalares $c_{j,i}$, $c_{j+1,i}$, ... , $c_{j+(m-1),i}$ se definen como variables aleatorias independientes y uniformes discretas cuyo conjunto soporte son los números enteros del 70 al 100 incluidos. Los demás escalares se definen como variables aleatorias independientes y uniformes discretas cuyo conjunto soporte son los números enteros del 1 al 30 incluidos. Este procedimiento se realiza para cada i desde 1 hasta q. Hay que tener presente que cada variable original debe tener una fuerte presencia en una única variable latente.

Como ejemplo ilustrativo supongamos que $p = 8, q = 3$ y que $U_d(a, b)$ es la distribución uniforme discreta discreta cuyo conjunto soporte son los números enteros desde a hasta b incluidos ($a, b \in \mathbb{N}, a < b$). Para la primera combinación lineal:

$$y_1 = c_{1,1}x_1 + c_{2,1}x_2 + c_{3,1}x_3 + c_{4,1}x_4 + c_{5,1}x_5 + c_{6,1}x_6 + c_{7,1}x_7 + c_{8,1}x_8 \tag{2.3.2}$$

Si se desea que x_1, x_2, x_3, x_4 estén representadas en y_1, entonces:

$$c_{1,1}, c_{2,1}, c_{3,1}, c_{4,1} \sim U_d\left(70,100\right) IID \tag{2.3.3}$$

$$c_{5,1}, c_{6,1}, c_{7,1}, c_{8,1} \sim U_d\left(1,30\right) IID \tag{2.3.4}$$

Para la segunda combinación lineal:

$$y_2 = c_{1,2}x_1 + c_{2,2}x_2 + c_{3,2}x_3 + c_{4,2}x_4 + c_{5,2}x_5 + c_{6,2}x_6 + c_{7,2}x_7 + c_{8,2}x_8 \tag{2.3.5}$$

Si se desea que x_5, x_6, x_7 estén representadas en y_2, entonces:

$$c_{5,2}, c_{6,2}, c_{7,2} \sim U_d\left(70,100\right) IID \tag{2.3.6}$$

$$c_{1,2}, c_{2,2}, c_{3,2}, c_{4,2}, c_{8,2} \sim U_d\,(1{,}30)\ IID \tag{2.3.7}$$

Finalmente, para la tercera combinación lineal:

$$y_3 = c_{1,3}x_1 + c_{2,3}x_2 + c_{3,3}x_3 + c_{4,3}x_4 + c_{5,3}x_5 + c_{6,3}x_6 + c_{7,3}x_7 + c_{8,3}x_8 \tag{2.3.8}$$

Nos queda únicamente la variable original x_8. Esta debe estar representada en y_3. Entonces:

$$c_{8,3} \sim U_d\,(70{,}100) \tag{2.3.9}$$

$$c_{1,3}, c_{2,3}, c_{3,3}, c_{4,3}, c_{5,3}, c_{6,3}, c_{7,3} \sim U_d\,(1{,}30)\ \ IID \tag{2.3.10}$$

En el ejemplo mencionado, las cuatro primeras variables originales tienen fuerte presencia en la primera variable latente, las tres siguientes variables originales lo hacen en la segunda variable latente y, la última variable original, tiene una fuerte representación en la tercera y última variable latente. Indicaremos esto, de manera general, mediante la sucesión finita $\{r_k\}_{k=1}^{q}$, cuyos elementos para el caso particular anterior son $r_1 = 4, r_2 = 3, r_3 = 1$.

El algoritmo diseñado e implementado simula una matriz $\mathbf{X}$ con n individuos y p variables originales. La matriz $\mathbf{A}$ de "scores" es de tamaño $n \times q$. Sin pérdida de generalidad, se supone que las entradas de dicha matriz son variables aleatorias independientes con distribución uniforme continua en el intervalo $[-1,1]$. Para construir la matriz $\mathbf{B}$ de las cargas con tamaño $p \times q$ se parte de la matriz que se construye con los escalares de las respectivas combinaciones lineales arriba indicadas. Esta matriz se nota por $\mathbf{C}$, y se muestra en la tabla 2.2:

	$\boldsymbol{y_1}$	$\boldsymbol{y_2}$	$\boldsymbol{y_3}$
$\boldsymbol{x_1}$	$c_{1,1}$	$c_{1,2}$	$c_{1,3}$
$\boldsymbol{x_2}$	$c_{2,1}$	$c_{2,2}$	$c_{2,3}$
$\boldsymbol{x_3}$	$c_{3,1}$	$c_{3,2}$	$c_{3,3}$
$\boldsymbol{x_4}$	$c_{4,1}$	$c_{4,2}$	$c_{4,3}$
$\boldsymbol{x_5}$	$c_{5,1}$	$c_{5,2}$	$c_{5,3}$
$\boldsymbol{x_6}$	$c_{6,1}$	$c_{6,2}$	$c_{6,3}$
$\boldsymbol{x_7}$	$c_{7,1}$	$c_{7,2}$	$c_{7,3}$
$\boldsymbol{x_8}$	$c_{8,1}$	$c_{8,2}$	$c_{8,3}$

Tabla 2.2: Matriz C con los coeficientes de las combinaciones lineales

Para obtener **B** se aplica el proceso de ortonormalización de Gram-Schmidt a la matriz **C**. Finalmente: $\mathbf{X} = \mathbf{A}\mathbf{B}^T$

El algoritmo que construye la matriz aleatoria **X** de tamaño $n \times p$ debe, en términos generales, implementar la transformación φ definida de la siguiente manera:

$$\begin{gathered}\boldsymbol{\varphi} : \mathbb{N} \times \mathbb{N} \times \mathbb{N} \times \underset{q}{\mathbb{N}^q} \to M_{n \times p} \\ \left(n, p, q, \{r_k\}_{k=1}^{q}\right) \mapsto \boldsymbol{\varphi}\left(n, p, q, \{r_k\}_{k=1}^{q}\right)\end{gathered} \tag{2.3.11}$$

Sujeto a las restricciones $n \geq p$ y $q < p$. Además, se debe satisfacer que:

$$p = \sum_{k=1}^{q} r_k \tag{2.3.12}$$

La transformación φ devuelve una matriz **X** de tamaño $n \times p$ que tiene la estructura ad hoc mencionada, la cual esperamos que sea detectada por el algoritmo CBPSO CD.

2.4.2 Estudio de simulación I

Para el primer ejemplo de simulación se ejecutó la transformación $\boldsymbol{\varphi}(100, 8, 3, \{4, 3, 1\})$. Es decir, se tienen cien individuos y cuarenta variables originales. Además, la estructura latente va a estar conformada por tres componentes disjuntas, la primera de ellas va a tener cargas diferentes de cero en las cuatro primeras posiciones. La segunda componente disjunta va a tener cargas diferentes de cero en las posiciones 5, 6 y 7. Y la última componente disjunta va a tener una carga diferente de cero en la última posición. Se obtuvo una matriz de datos **X** de tamaño 100×8. El algoritmo CBPSO DC entregó la matriz de cargas presentada en la tabla 2.3 y su representación gráfica en la figura 2.1:

	$\boldsymbol{y_1}$	$\boldsymbol{y_2}$	$\boldsymbol{y_3}$
$\boldsymbol{x_1}$	0.43413655	0	0
$\boldsymbol{x_2}$	0.53903007	0	0
$\boldsymbol{x_3}$	0.56700392	0	0
$\boldsymbol{x_4}$	0.44663027	0	0
$\boldsymbol{x_5}$	0	0.57919604	0
$\boldsymbol{x_6}$	0	0.56750625	0
$\boldsymbol{x_7}$	0	0.58520817	0
$\boldsymbol{x_8}$	0	0	1
% VAR	35.54%	33.01%	29.26%

Tabla 2.3: Matriz de cargas estudio de simulación I

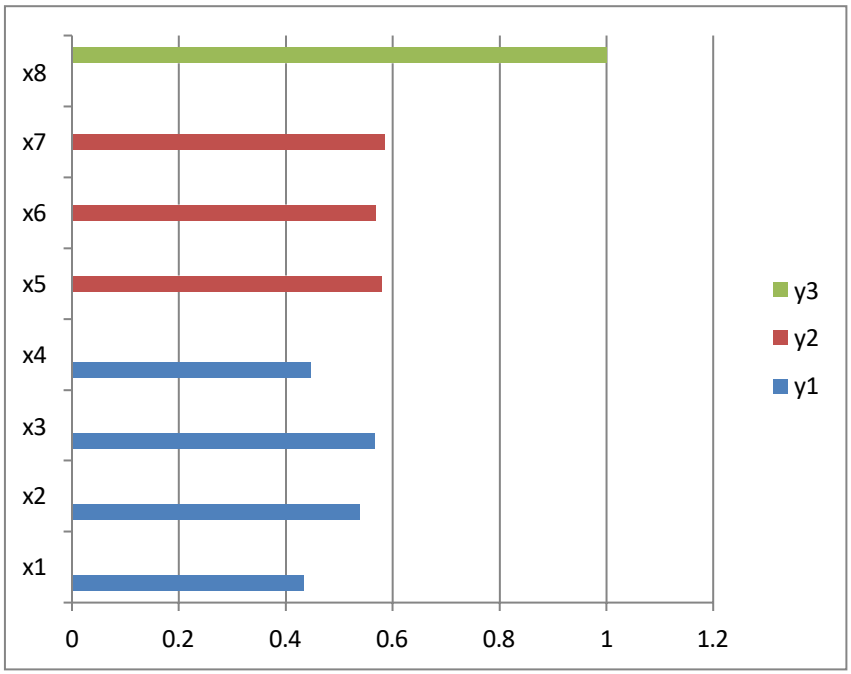

Figura 2.1: Gráfico de los valores de cargas estudio de simulación I

Se ejecutó cien veces el algoritmo CBPSO DC con cincuenta partículas y diez iteraciones como criterio de parada. En todas las cien ejecuciones se alcanzó la misma solución, mostrada en la tabla 2.3, demorándose dos o tres iteraciones a lo mucho. El algoritmo CBPSO DC pudo detectar la estructura ad hoc contenida en los datos. El fit que se obtuvo al aplicar componentes principales disjuntos es de 0.021859878 y la varianza total explicada por el modelo es de 97.81%. Igualmente se ejecutó cien veces el algoritmo DPCA y se alcanzó la misma solución dada en la tabla 2.3 en el 100% de las veces.

También se realizó un PCA clásico sobre la misma matriz de datos y se obtuvo la matriz de cargas dada en la tabla 2.4:

	y_1	y_2	y_3
x_1	0.39512468	-0.09020236	0.14824807
x_2	0.39775809	-0.00322755	0.40788776
x_3	0.44254179	-0.17190775	0.31755350
x_4	0.35239424	-0.11448361	0.25061667
x_5	0.26682407	0.46011771	-0.25159988
x_6	0.30662152	0.39441186	-0.26207760
x_7	0.35705213	0.28282245	-0.38998176
x_8	-0.27007773	0.70847497	0.60326462
% VAR	39.81%	32.27%	27.92%

Tabla 2.4: Matriz de cargas PCA clásico

Se observa que el PCA clásico no detecta de forma clara la estructura subyacente en los datos. Por ejemplo, es de resaltar, lo que ocurre con la segunda variable original x_2 donde no se puede concluir con certeza en cuál componente tiene fuerte presencia. Lo mismo se puede afirmar de la última variable original x_8.

2.4.3 Estudio de simulación II

Este ejemplo se diseñó para comparar el desempeño de los algoritmos CBPSO DC y el algoritmo DPCA cuando el tamaño de la matriz es mayor. El algoritmo de simulación implementó la transformación $\boldsymbol{\varphi}(200, 200, 3, \{50, 70, 80\})$ y se obtuvo una matriz de datos **X** de tamaño 200×200 y tres componentes disjuntas. Después de ejecutar ambos algoritmos cien veces cada uno, se resumen los resultados obtenidos en la tabla 2.5:

	CBPSO DC	DPCA
TIEMPO PROMEDIO DE EJECUCIÓN (MIN)	4.6	10.2
MEJOR TIEMPO DE EJECUCIÓN (MIN)	4.2	9.8
PEOR TIEMPO DE EJECUCIÓN (MIN)	4.8	10.4
TASA DE ÉXITOS	100%	93%

Tabla 2.5: Tabla resumen estudio de simulación II

Ambos algoritmos encontraron la mejor solución, con un fit de 0.02543703 y una varianza total explicada del 97.46%, con diferentes tasas de éxito. El algoritmo CBPSO DC se ejecutó con los mismos parámetros de entrada que en los experimentos computacionales anteriores, y el mejor tiempo alcanzado fue de 4.2 minutos en seis iteraciones. Por otro lado, para el algoritmo DPCA el mejor tiempo alcanzado fue de 9.8 minutos en dos iteraciones. La última fila de la tabla nos dice que el DPCA quedó atrapado en un óptimos locales en siete de las cien ejecuciones.

Es importante resaltar que el algoritmo DPCA realiza una búsqueda exhaustiva, pues a partir de la matriz binaria **V** que genera inicialmente de forma aleatoria, empieza a mover por fila y por columna el número uno en todas las posiciones posibles. A mayor número de variables originales y de variables latentes, el mencionado algoritmo presentará, como consecuencia, tiempos mayores de procesamiento.

El algoritmo CBPSO DC se adapta mejor en matrices de gran tamaño, ya que tiene la flexibilidad de permitir un ajuste en sus parámetros que le permiten una mejor estrategia de búsqueda. El algoritmo DPCA tiene únicamente un parámetro, la tolerancia, que es utilizado como criterio de parada.

Respecto de los porcentajes de acierto, el algoritmo DPCA tiene una mayor probabilidad de quedar atrapado en óptimos locales, ya que inicia su procesamiento con una única matriz binaria **V**, la cual como se ha dicho es modificada mediante desplazamientos del “uno” fila y columna, no necesariamente una de estas matrices obtenidas conduce al óptimo.

Como se menciona en (Macedo & Freitas, 2015), el algoritmo puede quedar atrapado en diferentes óptimos locales y se recomienda ejecutarlo varias veces. Por otro lado, el algoritmo CBPSO DC inicia con una matriz **V** por cada partícula, y es la búsqueda conjunta e inteligente de las partículas, lo que permite que el CBPSO DC tenga una menor probabilidad de quedar atrapado en un óptimo local.

2.5 Aplicaciones a Validación de Cuestionarios

2.5.1 Aplicación I: Trait Meta-Mood Scale 24

Recientes estudios muestran la importancia de la inteligencia emocional en la utilización de estrategias de aprendizaje. Un cuestionario que mide este constructo es el denominado TMMS24 como se detalla (Vega-Hernández et al., 2017). Este cuestionario consta de un conjunto de veinte y cuatro preguntas para evaluar los estados emocionales, cada una de ellas medida en una escala tipo Likert de 1 (Nada de acuerdo) a 5 puntos (Totalmente de acuerdo). Este instrumento se desprende del Trait-Meta Mood Scale (TMMS) del grupo de investigación (Sanchez-Garcia et al., 2016). Se ha utilizado a propósito este cuestionario, pues es un cuestionario validado, en el que se distinguen claramente sus dimensiones latentes.

TMMS 24 contiene tres dimensiones o factores que son: *Atención emocional* (preguntas de P1 a P8) se refiere a la capacidad que tiene una persona de expresar sus sentimientos. *Claridad emocional* (preguntas de la P9 a P16) manifiesta qué tanto una persona conoce sus estados emocionales. *Reparación emocional* (preguntas de la P17 a P24) es la facultad que tiene una persona de regular sus estados emocionales. En este experimento computacional se aplicó el cuestionario TMMS24 a 249 estudiantes de la Universidad de Salamanca, España, y se procesaron los datos primero con PCA clásico y luego con los dos métodos disjoint tratados, DPCA y CBPSO DC.

En la tabla 2.6 se muestran las cargas proporcionadas por PCA clásico. Tal como están dados los valores de las cargas factoriales resulta ser una tarea muy complicada interpretar

cada una de las variables latentes del cuestionario TMMS24 en términos de las dimensiones preestablecidas, debido a que no hay valores notablemente altos en valor absoluto que los diferencien del resto.

	PCA1	PCA2	PCA3		PCA1	PCA2	PCA3		PCA1	PCA2	PCA3
P1	0.23254	-0.23237	0.08136	P9	0.22998	0.08135	-0.36558	P17	0.16413	0.33062	0.21872
P2	0.22928	-0.25397	0.09367	P10	0.25446	0.04687	-0.33838	P18	0.14701	0.34381	0.24822
P3	0.23245	-0.26459	0.15229	P11	0.23467	0.08685	-0.26817	P19	0.15706	0.30838	0.27075
P4	0.27546	-0.19912	0.04398	P12	0.18133	0.00459	-0.21184	P20	0.16961	0.33375	0.22150
P5	0.14383	-0.26921	0.13748	P13	0.20757	0.00405	-0.14915	P21	0.16925	0.17367	0.10622
P6	0.18181	-0.23168	0.17093	P14	0.22040	0.09600	-0.30302	P22	0.19467	0.09546	0.11455
P7	0.18510	-0.23702	0.22662	P15	0.20857	0.04957	-0.16494	P23	0.10024	0.08436	0.07860
P8	0.25175	-0.20199	0.16564	P16	0.21579	0.05754	-0.21777	P24	0.21486	0.19223	0.12857

Tabla 2.6: Componentes principales clásicas para cuestionario TMMS24

Al utilizar los métodos DPCA y CBPSO DC, la mejor solución encontrada resultó ser exactamente la misma y se presenta en la tabla 2.7, y su representación gráfica en la figura 2.2:

	COMP1	COMP2	COMP3		COMP1	COMP2	COMP3		COMP1	COMP2	COMP3
P1	0.35227	0	0	P9	0	0	-0.41062	P17	0	0.42247	0
P2	0.36616	0	0	P10	0	0	-0.42258	P18	0	0.42936	0
P3	0.38711	0	0	P11	0	0	-0.38266	P19	0	0.41784	0
P4	0.36503	0	0	P12	0	0	-0.28486	P20	0	0.42935	0
P5	0.31388	0	0	P13	0	0	-0.28216	P21	0	0.28538	0
P6	0.32747	0	0	P14	0	0	-0.38453	P22	0	0.24283	0
P7	0.34493	0	0	P15	0	0	-0.30018	P23	0	0.15940	0
P8	0.36605	0	0	P16	0	0	-0.32812	P24	0	0.33528	0

Tabla 2.7: Componentes principales disjuntas para cuestionario TMMS24

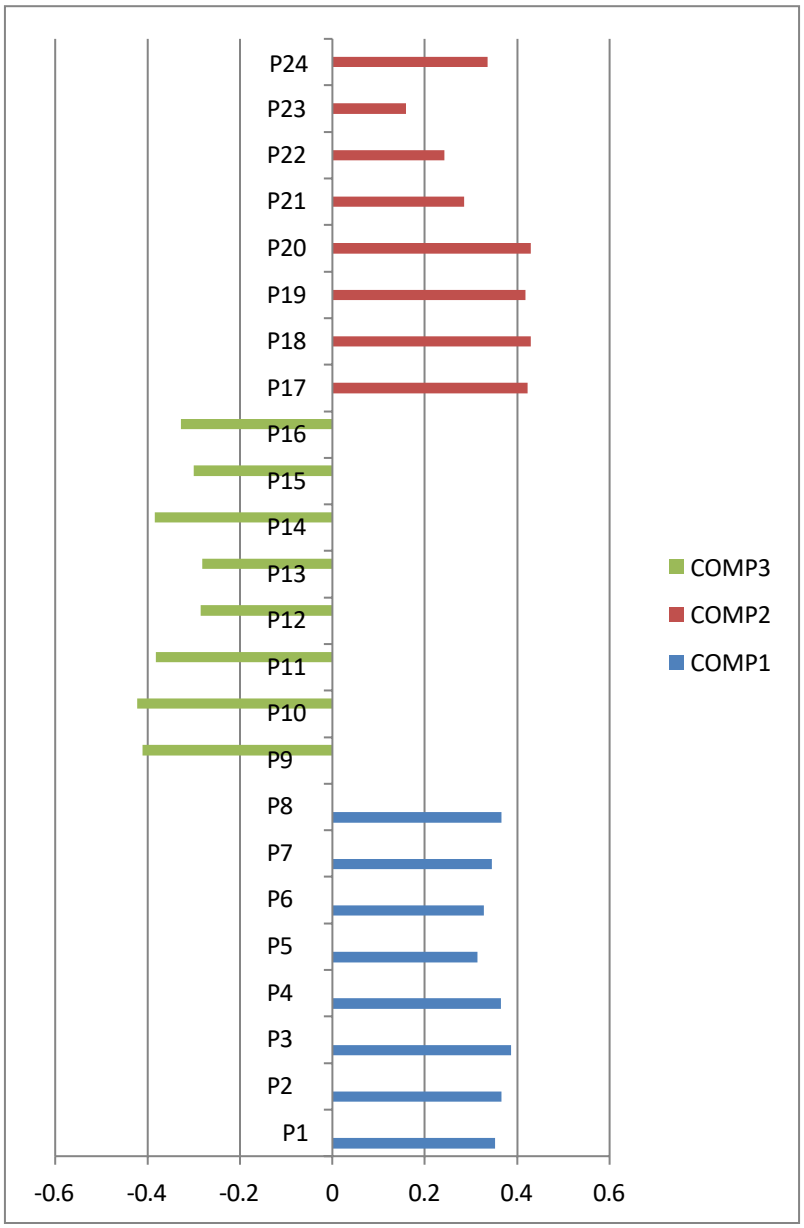

Figura 2.2: Gráfico de los valores de las cargas para cuestionario TMMS24

Se aprecia que los algoritmos disjoint pueden clasificar las veinte y cuatro preguntas del cuestionario TMMS24 en tres grupos claramente definidos, que corresponden precisamente a las dimensiones del cuestionario. Las preguntas P1 a P8 tienen presencia en el primer componente, que corresponde a “Atención emocional”. Las preguntas P9 a

P16 las podemos apreciar en el tercer componente, “Claridad de sentimientos”. Por último, las preguntas P17 a P24 se encuentran en el segundo componente, al que se lo identifica como “Reparación emocional”.

Los algoritmos disjoint distinguen de manera precisa las dimensiones que el cuestionario afirma tener, sin crear dimensiones espurias e inexistentes. Además, en los dos métodos disjoint el fit, o error relativo de la aproximación, para esta solución es igual a 0.4450664, mientras que para un PCA clásico es de 0.4306895. Hay una ligera pérdida de fit, pero a cambio se gana en interpretación. La proporción o porcentaje de varianza explicada por cada componente disjunta se presenta en la tabla 2.8 y resulta ser igual para ambos métodos comparados. Estos porcentajes se calcularon por medio de (2.2.19).

	COMP1	**COMP2**	**COMP3**	**TOTAL**
% VAR	20.45%	18.23%	16.81%	**55.49%**

Tabla 2.8: Porcentaje de varianza explicada por las componentes disjuntas

La proporción de varianza explicada por las tres componentes disjuntas consideradas es aproximadamente 1.5% menor que la varianza explicada por las tres primeras componentes principales calculadas de la forma clásica, como se puede advertir de la tabla 2.9.

	COMP1	**COMP2**	**COMP3**	**TOTAL**
% VAR	27.05%	19.27%	10.61%	**56.94%**

Tabla 2.9: Porcentaje de varianza explicada por PCA clásico

Donde se aprecia la mejora del algoritmo CBPSO DC con respecto al algoritmo DPCA es en la proporción de éxitos en alcanzar la mejor solución. El algoritmo CBPSO DC explora de manera más exhaustiva el conjunto de soluciones factibles lo que explica por qué no

queda atrapado en óptimos locales. Para cada uno de los métodos expuestos se realizaron cien ejecuciones. El método DPCA alcanzó la mejor solución (exhibida en la tabla 2.7) en el 98% de los casos, mientras que en el método CBPSO DC en el 100% de los casos.

2.5.2 Aplicación II: The General Self-Eficacy Scale

El concepto de autoeficacia general se refiere a la forma en que un individuo percibe su capacidad para realizar tareas nuevas o difíciles, y para hacer frente a dificultades. Este constructo se cuantifica a través de la General Self Efficacy Scale (GSE), ver (Scholz et al., 2002), establece que la estructura latente es unidimensional y universal. Es decir, que los diez ítems que conforman la escala conforman un factor que puede ser utilizado de manera generalizada en cualquier país. No obstante, como se señala en (Villegas et al., 2018) esta escala no es unidimensional ni universal, conclusión que se confirma mediante el análisis por componentes disjuntas presentado más adelante.

La escala GSE consta de 10 items evaluados con una escala de Likert de cuatro puntos, de acuerdo con la siguiente categorización: 1 "No del todo cierto", 2 "Casi cierto", 3 "Moderadamente cierto" y 4 "Exactamente cierto" (Schwarzer, R. & Jerusalem, 1995; Scholz et al., 2002). El significado de cada ítem se detalla en la tabla 2.10.

Item	Significado
Self1	Siempre puedo resolver problemas difíciles si me esfuerzo lo suficiente.
Self2	Si alguien se opone a mí, puedo encontrar los medios y formas de conseguir lo que quiero.
Self3	Es fácil para mí ceñirme a mis objetivos y lograr mis objetivos.
Self4	Confío en que podré afrontar con eficacia acontecimientos inesperados.
Self5	Gracias a mi ingenio, sé cómo manejar situaciones imprevistas.
Self6	Puedo resolver la mayoría de los problemas si invierto el esfuerzo necesario.
Self7	Puedo mantener la calma cuando me enfrento a las dificultades porque puedo confiar en mis capacidades hacer frente a las dificultades.
Self8	Cuando me enfrento a un problema, suelo encontrar varias soluciones.
Self9	Si estoy en problemas, normalmente puedo pensar en algo que hacer.
Self10	No importa lo que se me presente, por lo general soy capaz de manejarlo.

Tabla 2.10: Significado de las variables en GSE Scale

Para este ejemplo se utilizó la base de datos, con los 10 items y 19719 individuos, que está disponible en: http://userpage.fu-berlin.de/~health/selfscal.htm.

El algoritmo CBPSO DC muestra tres componentes disjuntos con un fit de 0.02820508 y una varianza total explicada de 58.93%. Este resultado es similar al obtenido por (Villegas et al. 2018). Se puede observar en la tabla 2.11, que las variables Self1 y Self6 se encuentran en el segundo componente. La variable Self2 es la única que se encuentra en el tercer componente. Las demás variables se encuentran en el primer componente. La información que aportan estos ítems no es recopilada en una única variable latente, contrario a lo que afirman los autores de la escala. Esta información se presenta de manera gráfica en la figura 2.3. Es importante concluir entonces que la escala GSE no es unidimensional, no se puede describir a través de un único factor.

	COMP1	COMP2	COMP3
Self1	0	0.71776986	0
Self2	0	0	1
Self3	0.36286421	0	0
Self4	0.36971158	0	0
Self5	0.37589985	0	0
Self6	0	0.69628042	0
Self7	0.38156436	0	0
Self8	0.38424172	0	0
Self9	0.3867569	0	0
Self10	0.38409408	0	0
% EV	35.4%	12.8%	10.73%

Tabla 2.11: Componentes disjuntos GSE con el algoritmo CBPSO

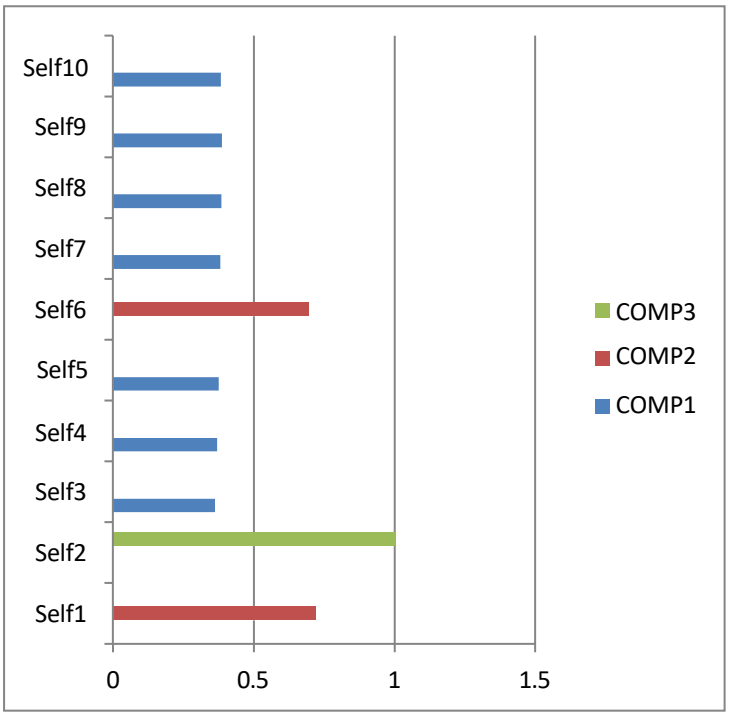

Figura 2.3: Gráfico de los valores de las cargas cuestionario GSE

Esta matriz de datos de gran tamaño también fue analizada con el algoritmo DPCA. Después de realizar cien ejecuciones con ambos algoritmos, los resultados obtenidos se detallan en la tabla 2.12:

	CBPSO DC	**DPCA**
TIEMPO PROMEDIO DE EJECUCIÓN (MIN)	2.55	0.09
MEJOR TIEMPO DE EJECUCIÓN (MIN)	1.55	0.07
PEOR TIEMPO DE EJECUCIÓN (MIN)	3.23	0.17
TASA DE ÉXITOS	100%	57%

Tabla 2.12: Resumen de resultados con datos GSE

Ambos algoritmos encontraron la mejor solución que es la presentada en la tabla 2.11, con diferentes tasas de éxito, tabla 2.12. El algoritmo CBPSO DC se ejecutó con 500 partículas, 30 iteraciones como máximo, 1.5 de peso social y peso cognitivo, y una inercia que varía linealmente de 0.5 a 3. El mejor tiempo alcanzado por el algoritmo CBPSO DC fue de 1.55 minutos en 4 iteraciones. Por otro lado, para el algoritmo DPCA el mejor tiempo alcanzado fue de 0.07 minutos en una iteración. La última fila de la tabla 2.12 indica que el DPCA quedó atrapado en óptimos locales en 43 de las 100 ejecuciones. Es importante señalar que debido a los valores ingresados en los parámetros del CBPSO DC, el tiempo de ejecución es mayor respecto del algoritmo DPCA, pero se obtiene como beneficio el 100% de eficacia. Esta flexibilidad no la tiene el algoritmo DPCA.

2.6 Una aplicación del algoritmo CBPSO DC a selección de variables en un estudio medio ambiental

Un problema que se presenta cuando se realizan estudios con datos medioambientales o ecológicos es el de la presencia de variables redundantes, es decir, variables que son combinación lineal exacta o aproximada de una o más de las variables restantes. Estas variables pueden ser eliminadas sin perder información o con una mínima pérdida de ella, resultando un conjunto de datos más pequeño lo que redunda en la interpretación de los resultados como en la reducción de los costos de dichos estudios.

Eliminar las variables redundantes, y retener a aquellas que más información aporten es de suma importancia, en particular si el conjunto de datos es masivo, como suele suceder en los estudios sobre medio ambiente. PCA permite reducir la dimensionalidad de un conjunto de datos. Las componentes principales al no estar correlacionadas eliminan la información redundante. El problema que se presenta, como ya se ha indicado, es su interpretación, debido a que las cargas no siempre se acercan a cero en valor absoluto. CBPSO DC permite paliar este problema presentando a las variables que contribuyen con poca información (a determinada componente principal) con valores de cero. De esta manera podemos seleccionar las variables que más información aportan a la primera componente principal disjunta (es decir, la que explica la mayor variabilidad) y descartar las variables que se encuentran en las componentes principales disjuntas restantes. Obviamente, esta decisión depende de muchos factores como son la clase de investigación a realizarse, de la importancia de las variables a excluirse y del criterio del investigador.

Existe una amplia bibliografía en la que se presentan diferentes metodologías para la reducción del número de variables en un estudio multivariante, entre las cuales podemos destacar: el trabajo seminal de (I. T. Jolliffe, 1973) que utiliza PCA, (Beale et al., 1967) mediante regresión lineal múltiple, y el trabajo de (King & Jackson, 1999) que es más específico, puesto que propone distintos métodos para descartar variables redundantes en estudios sobre medio ambiente. En el artículo de King y Jackson se muestra que el método Broken-stick, es el mejor criterio para determinar el número de componentes principales a

retener, para luego seleccionar las variables a descartar mediante los distintos modelos desarrollados por Jolliffe.

Cuando se reduce la dimensión de un conjunto de datos se presenta el problema de la estabilidad, que consiste en la concordancia entre las distancias de los individuos en el espacio generado por las variables originales y el espacio generado por el conjunto que contiene a las variables seleccionadas (Sorzano et al., 2014). Para que la reducción del número de las variables originales, mediante PCA, en datos ecológicos sea estable se recomienda, ver (Grossman et al., 1991), que la razón entre variables originales y variables seleccionadas no sea mayor a 3.

En esta sección se propone utilizar el método Broken-stick junto con el algoritmo CBPSO DC para seleccionar las variables que se deben retener en un estudio multivariante de datos ambientales.

Los datos utilizados se tomaron de la página de FOREGS-EuroGeoSurveys Geochemical Baseline Database (http://www.gtk.fi/publ/foregsatlas/) y corresponden a muestras de la concentración de minerales tomadas en el subsuelo de diferentes países de la Unión Europea. Esta base de datos bien se puede utilizar para medir el grado de contaminación del subsuelo. Los datos se encuentran en el archivo "C_XRF_data_2v6_8Feb06.xls" disponible en la dirección http://weppi.gtk.fi/ publ/foregsatlas/ForegsData.php, que se halla dentro de la página mencionada líneas arriba. Para el análisis se han eliminado las columnas correspondientes a las coordenadas de su ubicación geográfica y al país al que corresponden. La matriz de datos consta de 19 columnas o variables (los elementos y compuestos químicos) y 788 filas (locaciones donde fueron tomadas las muestras). Las variables representan las cantidades de los minerales encontradas en un Kg. de tierra del subsuelo en cada locación muestreada. Puesto que las unidades de medida difieren, dependiendo de la variable, vamos a utilizar variables estandarizadas para el análisis. La lista de variables se muestra en la tabla 2.13.

Variable	Chemical compound/element	Measurement units
SIO2_SI	Óxido de silicio	Kg.
TIO2_SI	Óxido de titanio	Kg.
AL2O3_SI	Óxido de aluminio	Kg.
FE2O3_SI	Óxido de hierro	Kg.
MNO_SI	Óxido de manganeso	Kg.
MGO_SI	Óxido de magnesio	Kg.
CAO_SI	Óxido de calcio	Kg.
NA2O_SI	Óxido de sodio	Kg.
K2O_SI	Óxido de potasio	Kg.
P2O5_SI	Óxido de fósforo	Kg.
BA_SI	Bario	mg.
CR_SI	Cromo	mg.
RB_SI	Rubidio	mg.
SN_SI	Estaño	mg.
SR_SI	Estroncio	mg.
W_SI	Tungsteno	mg.
Y_SI	Itrio	mg.
ZN_SI	Cinc	mg.
ZR_SI	Circonio	mg.

Tabla 2.13: Significado de las variables del estudio FOREGS

El procedimiento propuesto es el siguiente:

1. Aplicar un PCA clásico al conjunto de datos.

2. Seleccionar Q, es decir, el número de componentes principales a considerar mediante el modelo Broken-stick.

3. Calcular las Q componentes principales disjuntas con el algoritmo CBPSO DC.

4. Considerar a las variables que conforman la primera componente principal disjunta.

2.6.1 Aplicación:

Los resultados del PCA de la matriz de datos arrojaron los siguientes resultados, tabla 2.14:

PC	Valores propios	% VAR	% Varianza acumulada
1	5.095	26.815	26.815
2	2.941	15.477	42.292
3	1.838	9.674	51.966
4	1.488	7.833	59.799
5	1.301	6.845	66.644
6	1.025	5.397	72.041
7	0.904	4.759	76.800
8	0.797	4.193	80.993
9	0.722	3.801	84.794
10	0.650	3.423	88.217
11	0.489	2.572	90.789
12	0.412	2.166	92.955
13	0.387	2.039	94.994
14	0.325	1.708	96.703
15	0.267	1.408	98.110
16	0.125	0.657	98.768
17	0.105	0.554	99.322
18	0.079	0.417	99.739
19	0.050	0.262	100.000

Tabla 2.14: Componentes principales datos FOREGS

En la figura 2.4 se aprecia el gráfico correspondiente al método Broken-stick (línea azul) superpuesto a una curva en rojo que indica las proporciones de varianza explicadas al considerar las componentes principales indicadas en el eje X. El punto de corte está cercano a la tercera componente principal. Por esta razón vamos a considerar tres componentes principales a retener (es decir Q=3). Estas tres primeras componentes principales explican el 51.66% de la variabilidad total.

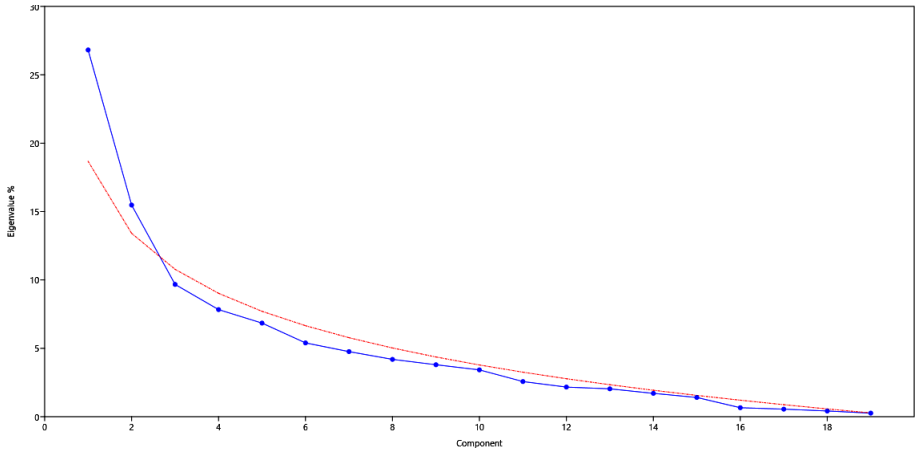

Figura 2.4: Diagrama Broken-stick datos FOREGS

Por tanto, para la aplicación del algoritmo CBPSO DC elegimos Q=3. Al aplicar el mencionado algoritmo se obtuvieron los resultados exhibidos en la tabla 2.15. En la figura 2.5 se presenta una representación gráfica de la tabla 2.15:

	COMP1	COMP2	COMP3
SIO2_SI	0	0	0.57973524
TIO2_SI	**0.44020387**	0	0
AL2O3_SI	0	-0.46399195	0
FE2O3_SI	**0.49562162**	0	0
MNO_SI	**0.44137161**	0	0
MGO_SI	0	0	-0.34305531
CAO_SI	0	0	-0.55125267
NA2O_SI	0	-0.28220297	0
K2O_SI	0	-0.49633089	0
P2O5_SI	**0.27387266**	0	0
BA_SI	0	-0.43570013	0
CR_SI	**0.18954184**	0	0
RB_SI	0	-0.47588306	0
SN_SI	0	-0.15767984	0
SR_SI	0	0	-0.34117431
W_SI	0	-0.13253821	0
Y_SI	**0.42150555**	0	0
ZN_SI	**0.27779775**	0	0
ZR_SI	0	0	0.35488125
% VAR	17.34%	16.86%	12.58%

Tabla 2.15: Componentes principales datos FOREGS

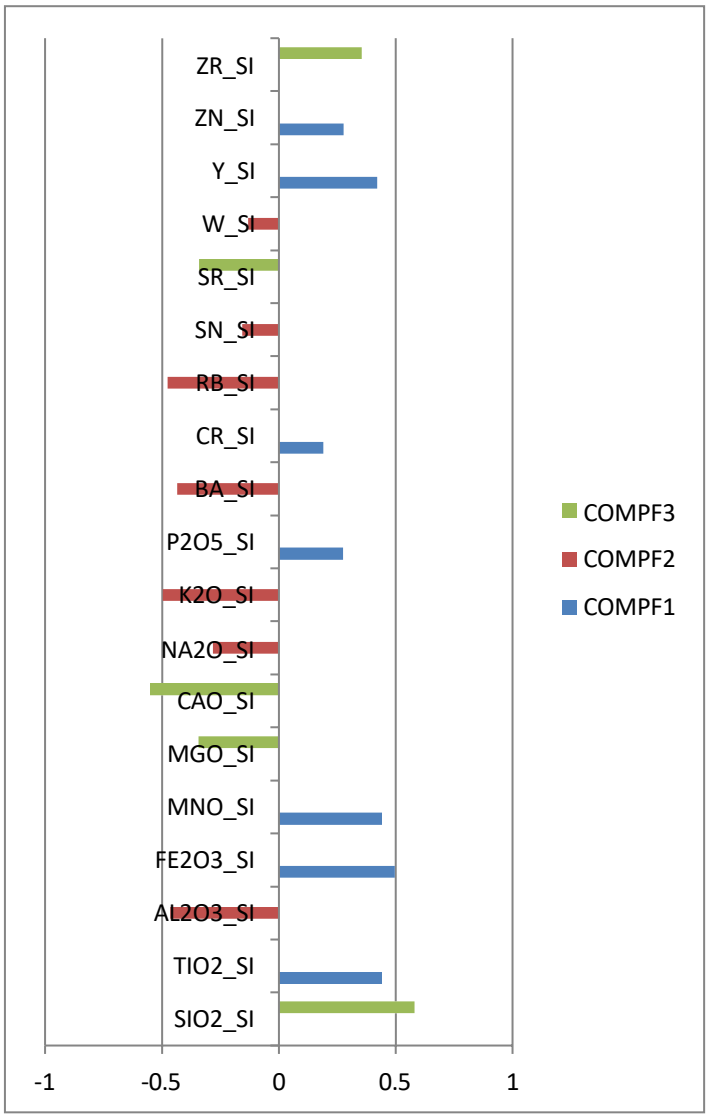

Figura 2.5: Gráfico de las componentes principales de los datos FOREGS

	% VAR	% Varianza acumulada
COMP1	17.34%	17.34%
COMP2	16.86%	34.20%
COMP3	12.58%	46.78%

Tabla 2.16: Porcentajes de varianza explicada por las componentes disjuntas

Las tres componentes disjuntas explican el 46.78% de la variabilidad total. Si consideramos las tres primeras componentes principales sugeridas por el método Broken-stick, que explican el 51.96% de la variabilidad, se tiene que las componentes disjuntas explican el 90.01% de la varianza explicada por las tres primeras componentes principales clásicas. Este menor porcentaje explicado por las componentes disjuntas se debe a que éstas deben satisfacer un conjunto de restricciones adicionales a las componentes principales clásicas.

El análisis de convergencia muestra que al algoritmo CBPSO DC le bastan 6 iteraciones para converger al valor 0.53220718.

Las variables a seleccionar, según nuestra propuesta, son las variables que constan en la primera componente principal disjunta COMP1 que es la de mayor porcentaje de varianza explicada. Hay siete variables con loadings diferentes de cero, que son: TIO2_SI, FE2O3_SI, MNO_SI, P2O5_SI, CR_SI, Y_SI, ZN_SI. La razón entre el número de variables originales y el número de variables seleccionadas es de 19/7 =2.71, que satisface la recomendación dada en (Grossman et al., 1991).

La aplicación del método CBPSO DC permite reducir de manera significativa el número de variables a considerarse en esta clase de estudios. De 19 variables originales, se reducen a sólo 7 variables. Por consiguiente, el gasto en reactivos, aparatos y personal se reduce también de forma considerable. Este tipo de análisis se puede aplicar a diversos problemas físico-químicos en los cuales se desee optimizar los gastos inherentes al proceso de investigación.

Capítulo 3

ANÁLISIS HJ BIPLOT DISJUNTO MEDIANTE OPTIMIZACIÓN POR ENJAMBRE DE PARTÍCULAS

Un biplot es una representación gráfica bidimensional de los datos contenidos en una matriz de tamaño *I*x*J*. El prefijo "bi" se refiere a que se muestran dos tipos de objetos: los individuos (filas) y las variables (columnas) en el mismo gráfico. Esta representación gráfica, que se fundamenta en la SVD de la matriz de datos, permite determinar patrones de interacción entre los individuos y las variables en un espacio de baja dimensionalidad.

Este capítulo se divide en las siguientes secciones:

3.1 Introducción

3.2 HJ Biplot clásico

3.3 Disjoint HJ Biplot por Optimización por Enjambre de Partículas (PSO DHJ Biplot)

3.4 Aplicación del Análisis PSO DHJ Biplot: Caso Covid Ecuador

3.1 Introducción

Los biplot fueron introducidos por Gabriel en 1971 (Gabriel, 1971) en el contexto de la representación gráfica del PCA. Tanto los individuos como las variables son representados como puntos en un espacio con producto interno de baja dimensionalidad como $\mathbb{R}^2$ o $\mathbb{R}^3$

Una revisión a fondo del desarrollo de los diferentes métodos biplot se encuentra en Nieto-Librero, 2015.

Consideremos una matriz de datos **X** de tamaño *I*x*J*, de rango $r \leq \min(I, J)$. La matriz **X** se puede factorizar de la forma:

$$\mathbf{X} = \mathbf{A}\mathbf{B}^T \tag{3.1.1}$$

Donde:

A es una matriz *I*x*r* y **B** es una matriz *J*x*r*.

Esta factorización no es única. Una forma de factorizar **X** es escoger *r* columnas de **A** como base ortonormal del espacio columna de **X**, y obtener $\mathbf{B} = \mathbf{Y}^T\mathbf{B}$.

En términos de los componentes de las matrices **X**, **A**, **B** se tiene:

$$x_{ij} = a_i^T b_j \tag{3.1.2}$$

Donde:

x_{ij} es el elemento que se encuentra en la fila *i* y en la columna *j* de la matriz **X**. a_i es la fila *i* de la matriz **A**, y b_j corresponde a la columna *j* de la matriz **B**. Los vectores fila y columna a_i y b_j son de dimensión *r*. Es decir, la matriz **X** que tiene *I*x*J* elementos se puede se puede expresar por medio de $I+J$ vectores de dimensión *r*.

Por ejemplo, si $I=30, J=10, r=2$, la matriz **X** tiene 300 elementos. Puesto que el rango es mucho menor que $J=10$, significa que la mayoría de las filas (columnas) son combinación lineal de únicamente dos de ellas. Utilizando la factorización $\mathbf{AB}^T$ para obtener **X**, se necesita disponer tan sólo de $(I+J)\times r = 60$ elementos, eliminando la información redundante. Con el añadido de que los individuos y columnas de la matriz **X** pueden representarse simultáneamente en el plano cartesiano. Si $r=3$, los 300 elementos de **X** se pueden obtener a partir de los 120 elementos contenidos en las matrices **A** y **B**, y sus elementos se pueden visualizar en un espacio tridimensional.

Una forma de obtener la factorización $\mathbf{X} = \mathbf{AB}^T$ es utilizando la descomposición en valores singulares (ver capítulo I):

$$\mathbf{X} = \mathbf{U\Lambda V}^T \tag{3.1.3}$$

Los elementos de **X** se pueden expresar:

$$x_{ij} = \sum_{k=1}^{\min(I,J)} \lambda_k u_{ik} v_{kj} \tag{3.1.4}$$

En general, el rango de la matriz **X** no va a ser igual a 2 o a 3, sino que $r \le \min(I,J)$. Si se utiliza un valor *K* menor al rango lo que se obtiene es una aproximación:

$$x_{ij} \simeq \sum_{k=1}^{K} \lambda_k u_{ik} v_{kj} \tag{3.1.5}$$

Sea $\mathbf{X}$ la matriz cuyos elementos están dados por $x_{ij} = \sum_{k=1}^{K} \lambda_k u_{ik} v_{kj}$. El error de aproximación K dimensional está dado por $\varepsilon_{ij} = x_{ij} - x_{ij}$ Para determinar la calidad de la aproximación se utiliza la norma de Frobenius, que corresponde a la aproximación por mínimos cuadrados:

$$\|\boldsymbol{\varepsilon}\|_F = \|\mathbf{X} - \mathbf{X}\|_F = \left(\sum_{i=1}^{I} \sum_{j=1}^{J} \left(x_{ij} - x_{ij} \right)^2 \right)^{\frac{1}{2}} \tag{3.1.6}$$

En (Eckart & Young, 1936) se demuestra que la mejor aproximación en el sentido de los mínimos cuadrados corresponde a la SVD.

Por tanto, al disponer de la descomposición en valores singulares $\mathbf{X} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{V}^T$ existen varias opciones para obtener la factorización $\mathbf{X} = \mathbf{A}\mathbf{B}^T$. Entre las más importantes tenemos:

GH Biplot: $\mathbf{A} = \mathbf{U}$, $\mathbf{B} = \mathbf{V}\boldsymbol{\Lambda}$, las filas se representan con máxima calidad de representación.

JK Biplot: $\mathbf{A} = \mathbf{U}\boldsymbol{\Lambda}$, $\mathbf{B} = \mathbf{V}$, las columnas se representan con máxima calidad de representación.

HJ Biplot: $\mathbf{A} = \mathbf{U}\boldsymbol{\Lambda}$, $\mathbf{B} = \mathbf{V}\boldsymbol{\Lambda}$ tanto filas como columnas se representan con máxima calidad de representación.

Los dos primeros biplot fueron dados por (Gabriel, 1971). La aplicación de la optimización por enjambre de partículas que se va a realizar es al HJ Biplot (Galindo, 1986).

3.2 HJ Biplot clásico

En los biplot tenemos dos tipos de objetos: los marcadores fila (filas de **A**) y los marcadores columna (columnas de **B**). Los primeros se representan como puntos y los segundos como vectores en el espacio dimensional reducido. Lo que permite proyectar las filas sobre la recta de acción de los marcadores columna.

La propuesta de representación gráfica del HJ Biplot admite incorporar en el mismo plano cartesiano los dos tipos de marcadores (filas y columnas) con la máxima calidad de

representación, es decir, con mínima distorsión. Permitiendo identificar de forma más clara la relaciones entre individuos y variables.

3.2.1 Interpretación del HJ Biplot:

Gracias a la aproximación obtenida mediante la SVD se tiene $x_{ij} \simeq \mathbf{a}_i^T \mathbf{b}_j$. Es decir, $x_{ij} \simeq \langle \mathbf{a}_i, \mathbf{b}_j \rangle$, de donde se tiene:

$$x_{ij} \simeq \left\| proy_{\mathbf{b}_j}^{\mathbf{a}_i} \right\| \left(\text{signo } \mathbf{b}_j \right) \left\| \mathbf{b}_j \right\| \tag{3.2.1}$$

$\left\| proy_{\mathbf{b}_j}^{\mathbf{a}_i} \right\|$ es la norma de la proyección de $\mathbf{a}_j$ sobre $\mathbf{b}_j$. Corresponde a la distancia entre el origen y el punto $\mathbf{a}_j$, figura 3.1.

signo $\mathbf{b}_j$ es el signo del producto interno $\langle \mathbf{a}_i, \mathbf{b}_j \rangle$

$\left\| \mathbf{b}_j \right\|$ corresponde a la distancia entre el origen y el punto $\mathbf{b}_j$, figura 3.1.

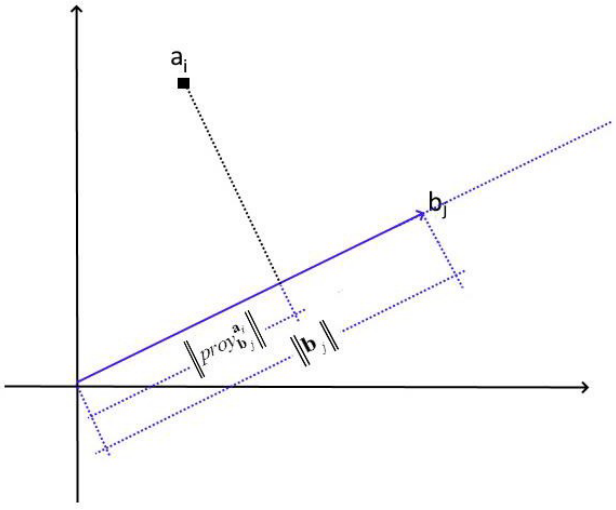

Figura 3.1: Marcadores fila y columna en el análisis Biplot

De las propiedades del producto interno estándar definido en el espacio dimensional reducido se deducen las siguientes propiedades del HJ Biplot:

1. La distancia entre los marcadores fila se puede interpretar como una medida de similaridad. Mientras más cercanos están más similares son.

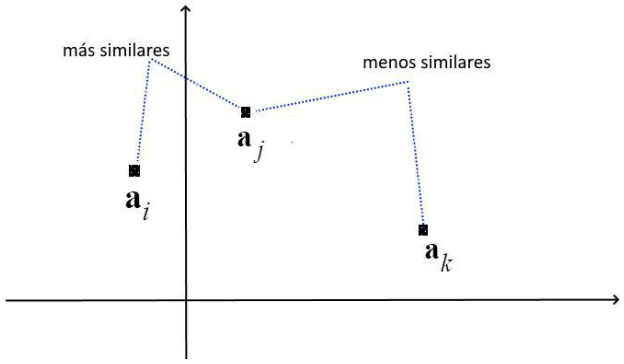

Figura 3.2: Similaridad en el HJ Biplot

2. El coseno del ángulo entre dos marcadores columna aproxima la correlación entre las variables correspondientes.

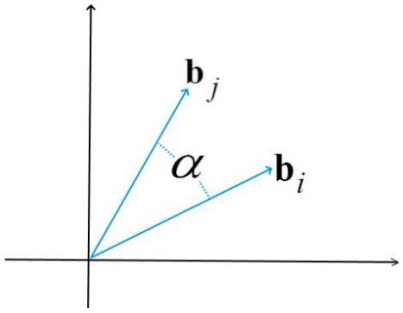

Figura 3.3: Correlación en el HJ Biplot

3. La norma de los marcadores columna aproxima la desviación estándar de las variables correspondientes.

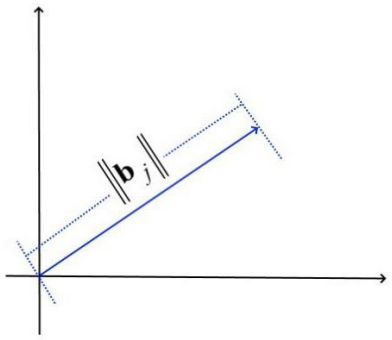

Figura 3.4: Aproximación de la desviación estándar en el HJ Biplot

4. Las proyecciones ortogonales de los marcadores sobre la recta de acción de un marcador columna cualquier marcan un orden respecto al origen que corresponde al valor medio de la variable asociada a dicho marcador columna. Así, mientras más alejada del origen esté la proyección de un individuo más estará más lejos de la media de esa variable. En la figura 3.5 el individuo correspondiente al marcador fila $\mathbf{a}_{i_2}$ es el que más se aleja de la media de la variable asociada al marcador columna $\mathbf{b}_j$.

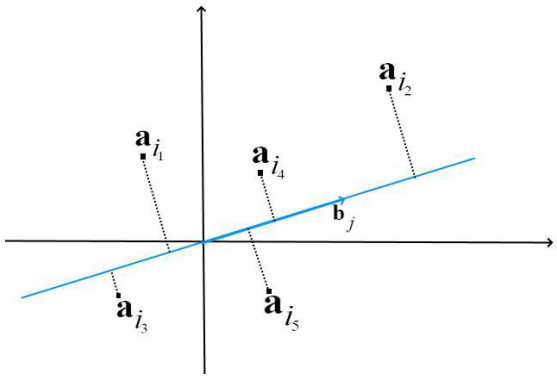

Figura 3.5: Proyecciones sobre los marcadores fila

Una de las desventajas propias de la naturaleza disjunta de los ejes principales es la de representar a las variables originales ya sea colineales o en ejes ortogonales. Por lo que se

debe tener en cuenta que en la proyección sobre el primer plano generado por los ejes disjuntos hay que interpretar con las debidas precauciones las verdaderas correlaciones entre las variables implicadas.

En el análisis multivariante en general y en el HJ Biplot en particular es fundamental determinar la o las variables que permitan caracterizar a los individuos en términos de sus diferencias, semejanzas y patrones de agrupaciones, para ello se utilizan las variables que más contribuyen a los ejes factoriales. Es en este punto donde es de primera importancia el determinar cuáles son las variables que más contribuyen a cada uno de los ejes, y para ello se tiene el denominado Disjoint HJ Biplot (Ana Nieto-Librero, 2015) que igual que en el PCA disjunto cada variable solo contribuye a un solo eje permitiendo una interpretación más sencilla de los ejes factoriales, y sobre todo, lo que es más importante en este caso, las posiciones relativas de los individuos en el plano de proyección.

3.3 Disjoint HJ Biplot mediante Optimización por Enjambre de Partículas (PSO DHJ Biplot)

Después de comprobar la eficiencia del método de optimización por enjambre de partículas aplicada al cálculo de las componentes principales disjuntas, en esta sección se va a adaptar al HJ Biplot, buscando más claridad en la interpretación de los ejes principales y por ende en la mejor explicación de los patrones y conglomerados que se pueden obtener en las soluciones que proporciona el HJ Biplot. Este nuevo algoritmo se denomina PSO DHJ Biplot.

En forma análoga a la metodología desarrollada para DPCA (capítulo 2), la metodología para el análisis HJ Biplot está basada en la factorización de la matriz de datos centrada $\mathbf{X}$ de tamaño $I \times J$:

$$\mathbf{X} = \mathbf{A}\mathbf{B}^T \tag{3.3.1}$$

Donde:

$\mathbf{A} = \lceil \lfloor a_{ij}$ matriz de tamaño $I \times Q$.
$\rfloor \rceil$

$\mathbf{B} = \lceil \lfloor b_{ij}$ matriz de tamaño $J \times Q$
$\rfloor \rceil$

La matriz **B** está sujeta a las siguientes restricciones propias de su carácter disjunto, tal como se explicó en el capítulo anterior, éstas son:

4 $\sum_{j=1}^{J} b_j^2 = 1, \quad q = 1,..,Q$

5. $\sum_{j=1}^{J} \left(b_{jq} b_{jr} \right)^2 = 0, \quad q = 1,..,Q-1; \quad r = q+1,..,Q$

6 $\sum_{q=1}^{Q} \lambda_j^2 > 0, \quad j = 1,..,J$

Puesto que Q es mucho menor que J se trata de una aproximación a bajo rango: $\mathbf{X} \simeq \mathbf{A}\mathbf{B}^T$ por tanto existe un error **E** propio de la aproximación:

$$\mathbf{X} = \mathbf{A}\mathbf{B}^T + \mathbf{E}, \tag{3.3.2}$$

De donde**:**

$$\mathbf{E} = \mathbf{X} - \mathbf{A}\mathbf{B}^T \tag{3.3.3}$$

Se tiene el mismo problema de minimización:

$$\begin{cases} \min F(\mathbf{A}, \mathbf{B}) = \left\| \mathbf{X} - \mathbf{A}\mathbf{B}^T \right\|^2 \\ \text{s.t } \mathbf{A}, \mathbf{B} \text{ como se describe arriba} \end{cases} \tag{3.3.4}$$

La solución a este problema de optimización está explicada en el capítulo 2. La diferencia radica en que se añade el cálculo de la matriz de valores propios $\mathbf{\Lambda}$.

En el paso cero del algoritmo DPCA a cada una de las matrices $\mathbf{W}_{0q}$ se la aplica la SVD. Obteniéndose Q SVDs:

$$\mathbf{W}_{0q} = \tilde{\mathbf{R}}\boldsymbol{\Lambda}\mathbf{T}^T$$

Con estas SVD se obtiene la matriz $\mathbf{B}_0$:

La columna q de $\mathbf{B}_0$ es el vector singular derecho correspondiente al mayor valor singular de la q-ésima SVD. De la misma forma podemos formar la matriz diagonal $\boldsymbol{\Lambda}$ con los valores propios: en la q-ésima posición de la diagonal de $\boldsymbol{\Lambda}$ ponemos el mayor valor propio correspondiente al mayor valor singular de la q-ésima SVD. Luego calculamos $\mathbf{A}_0$. A diferencia del DPCA donde $\mathbf{A}_0 = \mathbf{X}_0\mathbf{B}$ se tiene que $\mathbf{A}_0 = \mathbf{X}\mathbf{B}$ tal como establece la teoria del método HJ Biplot. Por último, se evalúa la función objetivo $F_0(\mathbf{A}_0, \mathbf{B}_0)$. Así lo hacemos en cada etapa k del algoritmo hasta obtener la convergencia. Al final se obtienen las matrices **B** y **A** necesarias en el HJ Biplot.

Al igual que el Disjoint PCA, se presenta el problema de calcular la varianza explicada por cada nuevo eje disjunto. La solución es la misma que se utilizó anteriormente. Es decir, calculando las varianzas de los individuos en el nuevo sistema referencial disjunto.

A continuación, se presenta el algoritmo para utilizado para obtener el Disjoint HJ Biplot clásico a partir de una matriz de datos centrada de tamaño IxJ minimizar. Este algoritmo es una aplicación directa del método de mínimos cuadrados alternantes.

Algoritmo Disjoint HJ Biplot clásico

1: Leer **X**, Q, nIter (número de iteraciones)

2: Inicializar $\mathbf{V}_0$, $k = 0$

3: Inicio de iteraciones

3.1 : Sea $k = k+1$

3.2 : Actualice $\mathbf{V}_k$

3.3 : Calcular la matriz de particiones $\mathbf{W}_{kq}$ para $q = 1,.,Q$

3.4 : Calcular la SVD de $\mathbf{W}_{kq}$ para $q = 1,.,Q$

3.5 : Obtener $\mathbf{B}_k$, $\mathbf{V}_k$, $\mathbf{A}_k$

3.6 : Actualizar $F_k = F(\mathbf{A}_k, \mathbf{B}_k)$

4: Repita las iteraciones hasta que $k =$nIter

5: Mostrar los resultados $\mathbf{A}_k$ y $\mathbf{B}_k$

Para implementar el método de optimización por enjambre de partículas se procede de igual forma que en el caso del DPCA. La diferencia está en el cálculo de la matriz $\mathbf{\Lambda}$ que permite obtener la matriz **A** tal como se describe en el HJ Biplot.

A continuación, se exhibe el algoritmo para obtener el PSO DHJ Biplot a partir de una matriz centrada por columnas **X**:

Algoritmo PSO DHJ Biplot

1: Leer **X**, Q, nIter (número de iteraciones)

2: Inicializar aleatoriamente P partículas

3: Sea finertia=maxIner

4: Para cada iteración $k = 1,.,$nIter

4.1 : Para cada partícula $p = 1,.,P$:

4.2 : $\mathbf{B}_p = \text{step}(\mathbf{B}_p, \mathbf{B}_p^*, \mathbf{B}^*, \text{wCognition}, \text{wSocial}, \text{finertia})$

4.3 : Si $F(\mathbf{B}_p) < F(\mathbf{B}_p^*)$, entonces $\mathbf{B}_p^* = \mathbf{B}_p$

4.4 : Si $F(\mathbf{B}_p) < F(\mathbf{B}^*)$, entonces $\mathbf{B}^* = \mathbf{B}$

4.5 : Con $\mathbf{B}_p$ calculamos $\mathbf{\Lambda}_p$ y $\mathbf{A}_p = \mathbf{X}\mathbf{B}_p \mathbf{\Lambda}_p^{-1}$

Fin (para cada partícula p)

5: Mostrar los resultados $\mathbf{B}_p$, $\mathbf{\Lambda}_p$ y $\mathbf{A}_p$

3.4 Aplicación del Análisis PSO DHJ Biplot: Caso Covid Ecuador

La enfermedad denominada COVID-19 es una infección respiratoria aguda, que se originó en la ciudad de Wuhan en la provincia de Hubei en la China central. Rápidamente se convirtió en una pandemia que ha afectado tanto los sistemas sanitarios como a la economía de todos los países del mundo. En Ecuador el primer caso se detectó el 27 de febrero de 2020, y para finales de marzo la pandemia comenzó a asolar todas las ciudades del país, sobre todo las grandes ciudades. Saturó toda la red de salud pública y privada del país, produciéndose una de las más altas tasas de mortalidad de Latinoamérica por no decir del mundo.

En lo que va de la pandemia se han publicado muchos trabajos en los que se presentan análisis multivariantes de los datos generados por organismos dedicados a monitorear el desarrollo de la pandemia COVID. En (Romeu, 2020b) y (Romeu, 2020a) se aplica el análisis de componentes principales y el análisis discriminante para determinar las variables más relevantes para explicar el impacto de la pandemia COVID. Se utilizan cinco variables para el análisis para establecer cuáles son las más significativas para diferenciar la infección entre condados del estado de New York. Los datos utilizados son los disponibles a mayo de 2020. En (Ferreira, 2020) se analiza el caso brasileño. A más de las variables inherentes a la pandemia COVID se consideran variables socio-económicas como el PIB (Producto Interno Bruto) y el IDH (Índice de Desarrollo Humano) para realizar un análisis de conglomerados no jerárquico y un análisis factorial con el objetivo de determinar conglomerados de estados y las asociaciones entre las variables consideradas. Los datos son los disponibles a junio de 2020. (Devkota, 2021) presenta varios estudios estadísticos sobre los efectos de la pandemia COVID en Nepal. Inicia con una regresión logística multinomial, luego hace un ajuste temporal mediante modelos ARIMA, seguido de un modelo lineal de efectos mixtos, para terminar con PCA de

componentes principales para analizar cómo se interrelacionan las variables consideradas. En el documento (Mahmoudi et al., 2021) se realiza un PCA para clasificar a los países objeto del análisis (Francia, Alemania, Irán, Italia, España, Reino Unido y Estados Unidos) según su tasa de contagios. Los datos son tomados a abril de 2020. (Kumar et al., 2020) aplican un modelo estadístico exponencial para predecir la evolución temporal de los casos confirmados de COVID a través del tiempo en India, China, Estados Unidos, Alemania, Italia, Japón, Irán y Canadá. Además, realiza un análisis de correlación y de componentes principales, incluyendo variables socio-económicas, para identificar patrones en la propagación de la pandemia en los estados de la India. Utiliza datos a mayo de 2020. (Luo et al., 2020) realizan un PCA y un análisis factorial de los datos de síntomas clínicos obtenidos de pacientes con COVID en un hospital de China, el propósito del estudio es obtener correlaciones entre las variables consideradas que sirvan de base para la prevención y control de la pandemia. (Konishi, 2020) aplica las componentes principales para resaltar las diferencias y semejanzas entre virus de la gripe y el virus SARS COV2, causante de la pandemia COVID. Para este fin utiliza los datos de secuenciación del genoma de estos virus. En (Cobre et al., 2020) se investigan los factores de riesgo relacionados con la muerte de pacientes con COVID-19, mediante regresión logística. Los datos son tomados a abril de 2020. (Ye et al., 2020) aplican PCA y análisis de conglomerados para identificar fenotipos de la enfermedad COVID-19, que ayuden a la predicción de la severidad de la infección. Los datos son tomados a marzo de 2020.

Este trabajo tiene como objetivo explicar la dinámica de la epidemia del COVID-19 en el Ecuador desde un punto de vista multivariante, a partir de los datos recolectados a marzo de 2021. En particular, desde la perspectiva del análisis Disjoint HJ Biplot. La ventaja de los métodos multivariantes es la de considerar simultáneamente la información contenida en todas las variables consideradas, y no de cada variable por separado o por parejas mediante la correlación. Esto permite capturar las interrelaciones y patrones multidimensionales que se puedan presentar en la data.

Para un primer análisis se seleccionaron diez variables medidas sobre los 221 cantones en que se subdividen las 24 provincias de la república del Ecuador. Las variables consideradas son:

POB: el número de habitantes de cada cantón.

TASA: es la tasa de contagio corresponde al número de personas contagiadas por cada 100.000 habitantes.

CASOS: el número total de casos acumulados desde el inicio de la pandemia hasta mediados de marzo de 2021.

PROD: es el valor generado, por cada cantón, al fabricar bienes o proporcionar servicios. Se mide en millones de dólares.

EDU: es el valor invertido, en cada cantón, para cubrir los gastos en educación pública y privada. Se mide en millones de dólares.

ELEH: es la inversión del estado en proporcionar servicios de electricidad y agua en cada cantón. Se mide en millones de dólares.

VAB: es el Valor Agregado Bruto (VAB), una variable macroeconómica que muestra el aporte que adquieren los bienes y servicios al ser transformados durante el proceso productivo en cada cantón. Se mide en millones de dólares.

POBR: es el porcentaje de habitantes del cantón que vive en condiciones de pobreza.

Para un segundo análisis se consideran sólo los cantones cuya población supera los 90 mil habitantes, puesto que para estos cantones se dispone de la variable **EXCM** que corresponde al exceso de personas fallecidas por cada 100 mil habitantes a lo largo de la pandemia. Para este análisis se toma en cuenta a más de las anteriores variables a la variable **EXCM**.

Se dispone también de las variables: **HOSP** el número de hospitales públicos y privados en cada cantón. **CDS** el número de centros de salud y dispensarios médicos públicos y

privados en cada cantón. Pero no van a ser utilizadas en el presente trabajo puesto que no son proporcionales a las poblaciones donde se encuentran y desvirtúan los resultados obtenidos, ver (Romeu, 2020b).

Los datos se obtuvieron de las siguientes páginas web: Instituto Nacional de Estadística y Censos, https://www.ecuadorencifras.gob.ec/estadisticas/; página del Banco Central del Ecuador, https://contenido.bce.fin.ec/documentos/Estadisticas/SectorReal/CuentasCantonales/Indice.htm; página del sistema Geosalud 3.7.7 del Ministerio de Salud Pública del Ecuador, https://geosalud.msp.gob.ec/geovisualizador/index.php; y Observatorio social del Ecuador https://www.covid19ecuador.org/cantones. En la tabla 3.1 se listan las provincias del Ecuador y su número de cantones. En la figura 3.6 se exhibe un mapa de la división en provincias del país con sus respectivos cantones.

	Provincia	Número de cantones
1	Azuay	15
2	Bolívar	7
3	Cañar	7
4	Carchi	6
5	Cotopaxi	7
6	Chimborazo	10
7	El Oro	14
8	Esmeraldas	7
9	Guayas	25
10	Imbabura	6
11	Loja	16
12	Los Rios	13
13	Manabí	22
14	Morona Santiago	12
15	Napo	5
16	Pastaza	4
17	Pichincha	8
18	Tungurahua	9
19	Zamora Chinchipe	9
20	Galápagos	3
21	Sucumbíos	7
22	Orellana	4
23	Santo Domingo	2
24	Santa Elena	3
Total		221

Tabla 3.1: Número de cantones por provincia

Figura 3.6: Mapa de las provincias del Ecuador con sus cantones

Ambos estudios se realizaron para dos y tres componentes disjuntas. Se presentan los resultados para dos componentes disjuntas puesto que la disminución de la varianza explicada es pequeña, respecto a considerar tres componentes, pero a cambio se gana en interpretación.

Los resultados que se obtuvieron son los siguientes:

En el **primer estudio**, 221 cantones, 8 variables y 2 componentes disjuntas:

Los resultados proporcionados por el análisis Disjoint HJ Biplot se muestran en la tabla 3.2, y su representación gráfica en la figura 3.7.

	COMP1	COMP2	
POB	14.6826	**0.0000**	
TASA	**0.0000**	-12.4240	
CASOS	13.4997	**0.0000**	
PROD	14.7263	**0.0000**	
EDU	14.6079	**0.0000**	
ELEH	13.9450	**0.0000**	
VAB	14.7562	**0.0000**	
POBR	**0.0000**	12.4240	**Total**
% VAR	70.1519	17.4609	87.6128

Tabla 3.2: Cargas, componentes principales disjuntas Estudio 1.

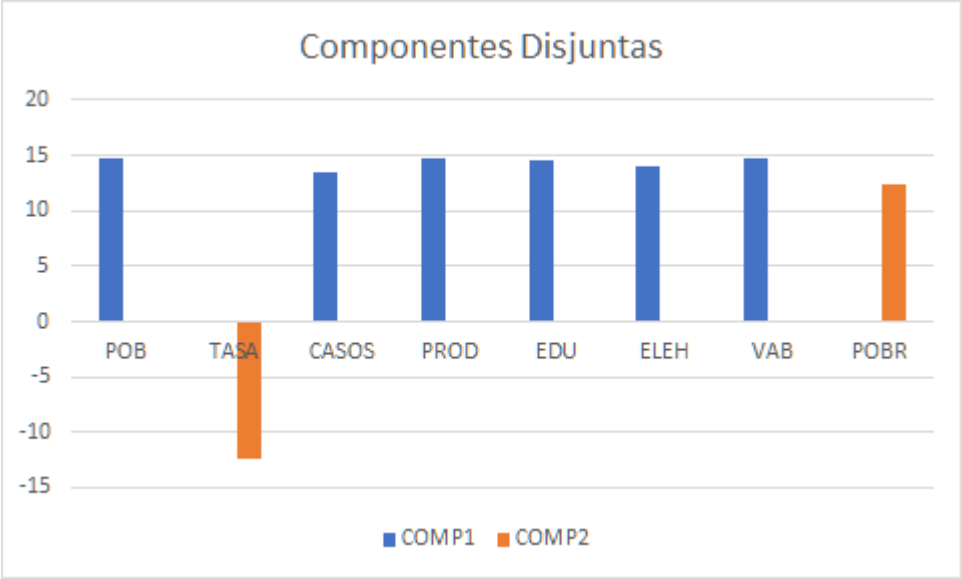

Figura 3.7: Gráfico de las componentes principales disjuntas Estudio 1.

En el caso del Disjoint HJ Biplot, las variables consideradas en el estudio o bien forman un ángulo de 0°/180° (esto es, están correlacionadas positivamente o negativamente) en caso de pertenecer a una misma componente disjunta, o bien un ángulo de 90° (no están correlacionadas) si pertenecen a diferentes componentes disjuntas. Estas correlaciones hay que tomarlas con mucha precaución. No corresponden a las correlaciones de las variables en el espacio original en el cual están definidas. El Disjoint HJ Biplot realiza una proyección de los datos originales sobre un espacio bidimensional, cuyos ejes están dados

por las componentes disjuntas, por ende, hay una deformación (igual que sucede en cualquiera de los análisis multivariantes, como por ejemplo PCA o cualquiera de los diferentes Biplots). En esta proyección aparecen las variables ya sea perteneciendo a uno u otro disjunto. Cosa que no sucede en el espacio original de las variables.

La variabilidad aproximada de cada variable viene dada directamente por el valor de la coordenada de la variable en la componente disjunta. Y se pueden calcular fácilmente tal como se indica en la teoría del PSO DHJ Biplot. Son un indicativo de que tanto se deformaron los datos al proyectarlos en el plano formado por los ejes disjuntos. Estos valores se detallan en la última fila de la Tabla 3.2. La varianza explicada por los dos ejes disjuntos considerados es del 87.61%, lo que es un valor bastante aceptable en el contexto multivariante.

La primera componente refleja la influencia del tamaño de la población. Las variables consideradas en esta componente son POB (población del cantón), CASOS (número de casos confirmados), PROD (el valor de la producción de bienes y servicios), EDU (presupuesto destinado a servicios educativos), ELEH (gasto en servicios de electricidad y agua potable), VAB (Valor Agregado Bruto). Todas estas variables dependen del tamaño de la población. La segunda componente contrapone las variables TASA (tasa de contagios) y POBR (porcentaje de la población que vive en condiciones de pobreza). Estas dos variables se diferencian de las variables que constan en la primera componente en que son porcentuales, es decir adimensionales. Además, nos brinda una contribución importante a la investigación: En el Ecuador la tasa de contagios está negativamente correlacionada con el nivel de pobreza. Lo que implica que mientras más bajo sea el nivel de pobreza mayor es la tasa de contagios. Este hecho se puede entender a que en el Ecuador las la población más pobre se encuentra en el campo, donde a más de estar dispersa hay pocas vías de comunicación. Las variables con la más alta variabilidad son las variables TASA y POBR, se encuentran asociadas a la segunda componente disjunta. Mientras que la variable que presenta la menor variabilidad es CASOS y se asocia a la primera componente principal.

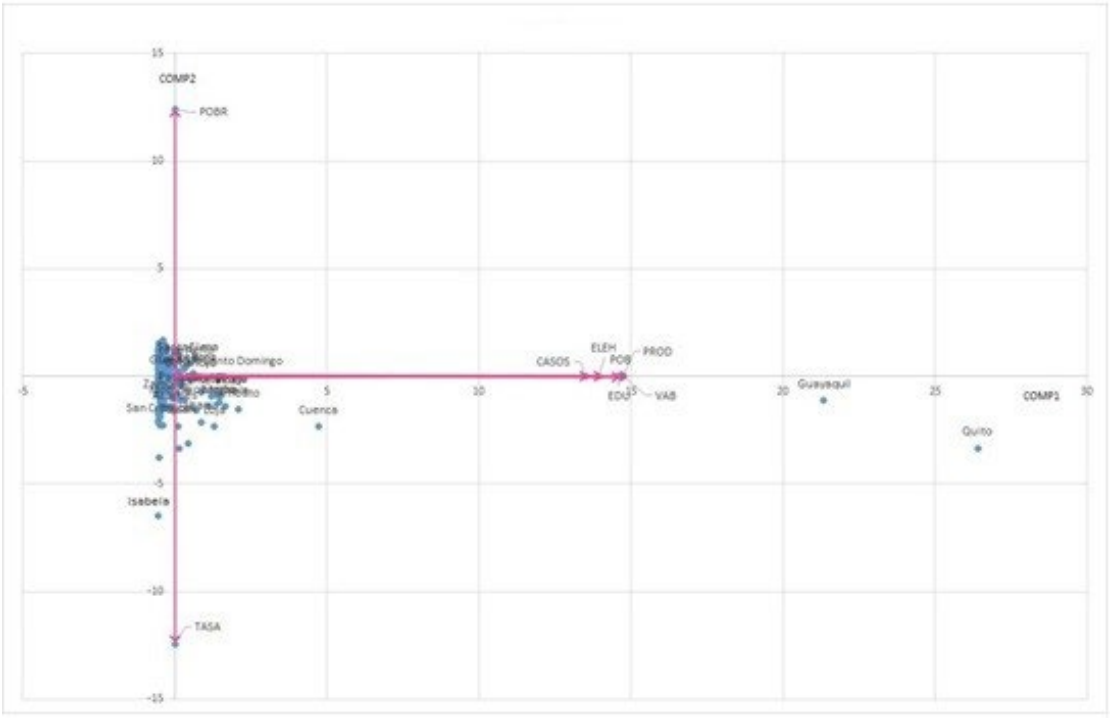

Figura 3.8: PSO DHJ Biplot Estudio 1.

En el biplot presentado en la figura 3.8, considerando la primera componente disjunta, se observa que el cantón Quito es el que se separa significativamente del resto. Está situación se debe a que es el cantón con la mayor población, también es el que mayor número de contagios ha presentado a lo largo de la pandemia. Seguido por los cantones Guayaquil y Cuenca que corresponden a cantones donde se encuentran las mayores ciudades del Ecuador, de nombres homónimos. Siendo Guayaquil la segunda ciudad más poblada y Cuenca la tercera.

Ecuador está dividido geográficamente en cuatro regiones Costa, Sierra, Oriente y Región insular. Para despejar la nube de datos de la figura 3.8, a continuación, se presenta por regiones. El PSO DHJ Biplot de la región Costa se muestra en la figura 3.9, el El PSO DHJ

Biplot de la región Sierra en la figura 3.10, y el El PSO DHJ Biplot de la región oriental en la figura 3.11. La región Insular es muy pequeña en términos poblacionales y número de cantones, razón por la que no se grafica.

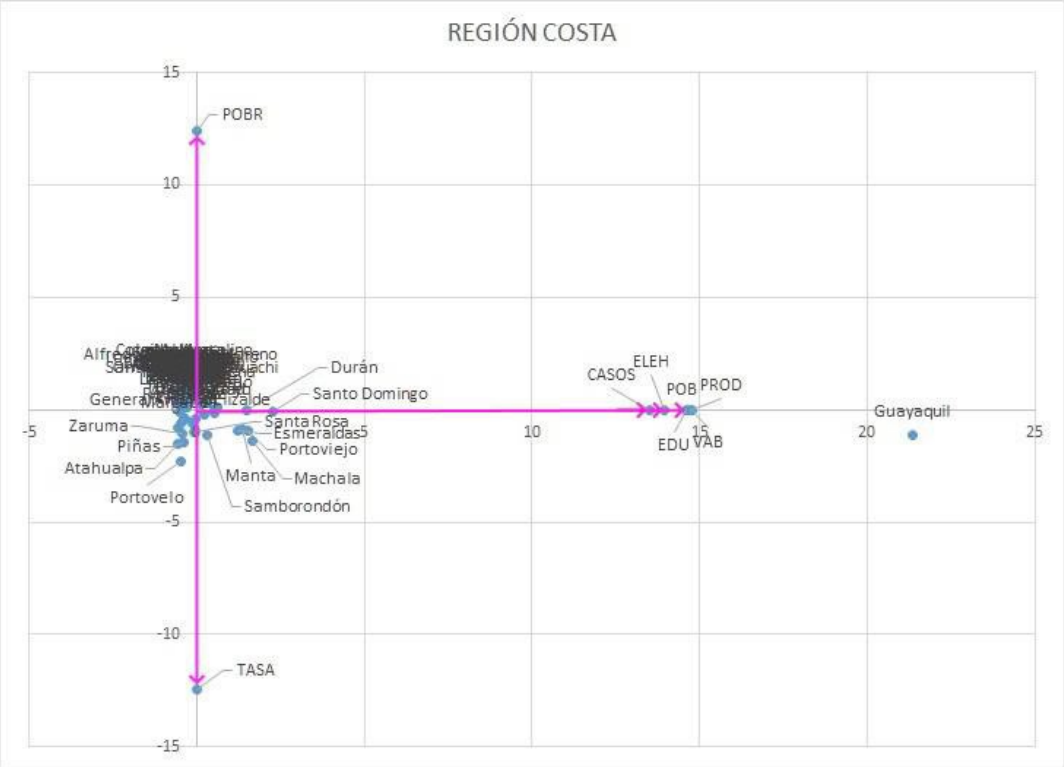

Figura 3.9: PSO DHJ Biplot Región Costa

En la región costa el cantón predominante en cuanto a la primera componente disjunta es Guayaquil. Mientras que respecto a la segunda componente se ubica en una posición cerca del origen de coordenadas, lo que indica que tanto su nivel de pobreza como la tasa de contagios son cercanos al promedio general. A excepción de Guayaquil, todo el resto de cantones de la región Costa forma un conglomerado en torno al origen. Se observa que el cantón Samborondón, su proyección en la segunda componente disjunta se ubica cerca a Guayaquil. Esto se puede explicar por el hecho de que conforma el llamado Gran Guayaquil, junto al cantón Durán, cuya proyección en la segunda componente también está cerca a Guayaquil.

Figura 3.10: PSO DHJ Biplot Región Sierra

En la región Sierra, respecto a la primera componente disjunta, el cantón que más destaca es Quito, seguido de lejos por Cuenca. Al igual que ocurre en la región Costa, el resto de cantones forma un conglomerado en torno al origen. La proyección en la segunda componente disjunta del cantón Rumiñahui está ligeramente más abajo de la del cantón Quito. En el cantón Rumiñahui se encuentran poblaciones satélites de Quito, y es un lugar de fincas y conjuntos habitacionales de las clases acomodadas de la ciudad capital, de allí que su nivel de pobreza sea menor al de Quito.

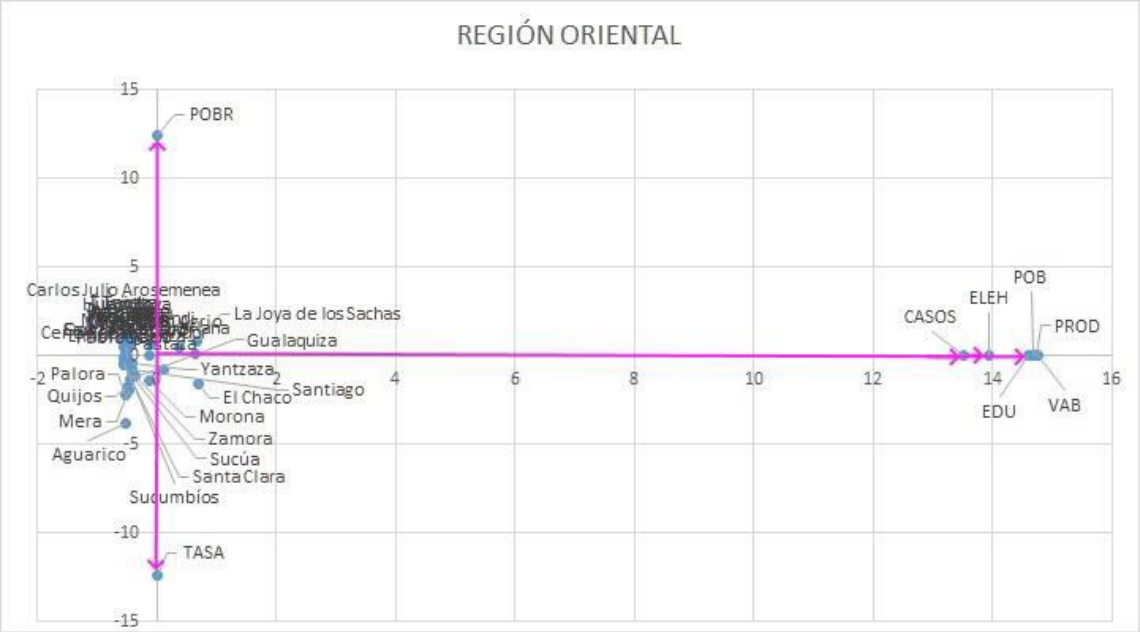

Figura 3.11**:** PSO DHJ Biplot Región Oriental

En la región Oriental no se presentan cantones que se separen del gran conglomerado central. Esto se debe a que todas las poblaciones orientales son pequeñas tanto en población como en recursos. Respecto al nivel de pobreza el cantón Aguarico es el que mejor se sitúa. En este cantón se encuentran las dos reservas naturales más importantes del país por su biodiversidad y ecoturismo.

En el **segundo estudio**, se consideraron 35 cantones, 9 variables y 2 componentes disjuntas, que se detallan en la tabla 3.3:

1	Ambato	8	El Carmen	15	Latacunga	22	Naranjal	29	Riobamba
2	Babahoyo	9	Esmeraldas	16	Loja	23	Orellana	30	Rumiñahui
3	Cayambe	10	Guaranda	17	Machala	24	Otavalo	31	Salinas
4	Chone	11	Guayaquil	18	Manta	25	Portoviejo	32	Samborondón
5	Cuenca	12	Ibarra	19	Mejía	26	Quevedo	33	Santa Elena
6	Daule	13	La Libertad	20	Milagro	27	Quinindé	34	Sto Domingo
7	Durán	14	Lago Agrio	21	Montecristi	28	Quito	35	Tulcán

Tabla 3.3: Cantones con más de 90 mil habitantes

En este estudio sólo se consideran los cantones con más de 90 mil habitantes y se añade la variable EXCM (exceso de personas fallecidas por cada 100 mil habitantes). El análisis PSO DHJ Biplot se realizó para dos componentes disjuntas. Se tiene los siguientes resultados, tabla 3.4, y su respectiva gráfica en la figura 3.12:

	COMP1	**COMP2**	
POB	5.8007	0.0000	
TASA	**0.0000**	-5.3071	
CASOS	5.1982	0.0000	
EXCM	2.3442	0.0000	
PROD	5.7977	0.0000	
EDU	5.7686	0.0000	
ELEH	5.7985	0.0000	
VAB	5.8099	0.0000	
POBR	**0.0000**	5.3071	**Total**
% VAR	63.6294	17.8830	81.5124

Tabla 3.4: Cargas, componentes principales disjuntas Estudio 2.

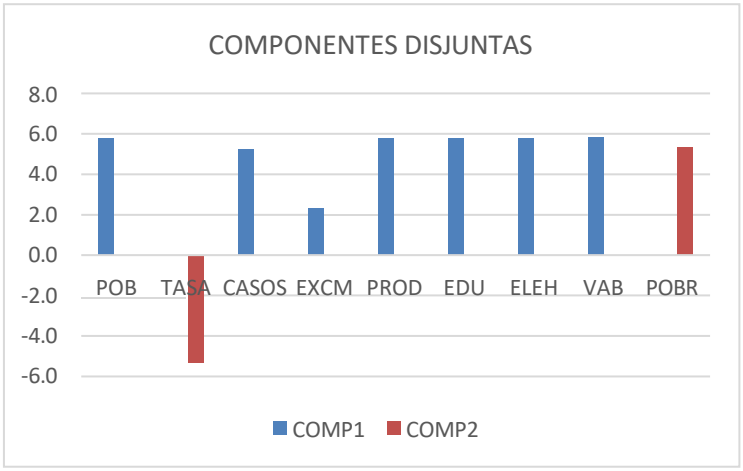

Figura 3.12: Gráfico de las componentes principales disjuntas Estudio 2.

En términos de las cargas de las componentes disjuntas la situación es similar al estudio 1. Con una disminución de la varianza explicada de un 6.1%. Las variables TASA y POBR conforman la segunda componente disjunta y el resto pertenecen a la primera componente. La nueva variable EXCM agregada se ubica en la primera componente disjunta con la más pequeña contribución (2.3442), ver tabla 3.4. La representación gráfica del PSO DHJ Biplot se exhibe en la figura 3.13.

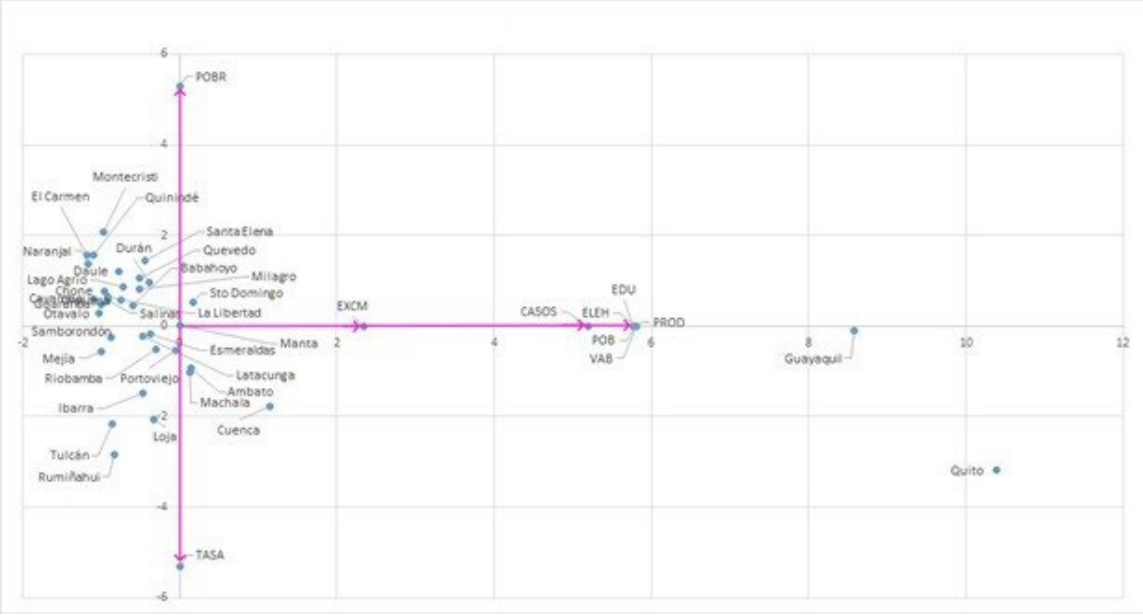

Figura 3.13: PSO DHJ Biplot Estudio 2.

Si se utiliza el segundo eje disjunto como un índice sintético de la dualidad Pobreza-Tasa de contagios. Podemos ordenar a los cantones en forma ascendente, en base a las proyecciones en este eje, tal como se muestra en la tabla 3.5 y en la figura 3.14:

	CANTÓN	**COMP2**		**CANTÓN**	**COMP2**		**CANTÓN**	**COMP2**
1	Quito	-3.1835	13	Latacunga	-0.2181	25	Chone	0.7712
2	Rumiñahui	-2.8428	14	Esmeraldas	-0.1833	26	Milagro	0.8287
3	Tulcán	-2.1687	15	Guayaquil	-0.1066	27	Lago Agrio	0.8592
4	Loja	-2.0602	16	Manta	0.0243	28	Durán	0.9617
5	Cuenca	-1.7789	17	Otavalo	0.2785	29	Quevedo	1.0591
6	Ibarra	-1.4919	18	Babahoyo	0.4449	30	Daule	1.2002
7	Machala	-1.0239	19	Guaranda	0.4783	31	Naranjal	1.3652
8	Ambato	-0.9275	20	Sto Domingo	0.5321	32	Santa Elena	1.4367
9	Mejía	-0.5545	21	Salinas	0.5666	33	Quinindé	1.5606
10	Portoviejo	-0.5403	22	Orellana	0.5691	34	El Carmen	1.5658
11	Riobamba	-0.5031	23	Cayambe	0.5977	35	Montecristi	2.0794
12	Samborondón	-0.2485	24	La Libertad	0.6528			

Tabla 3.5: Orden de los cantones según la segunda componente disjunta.

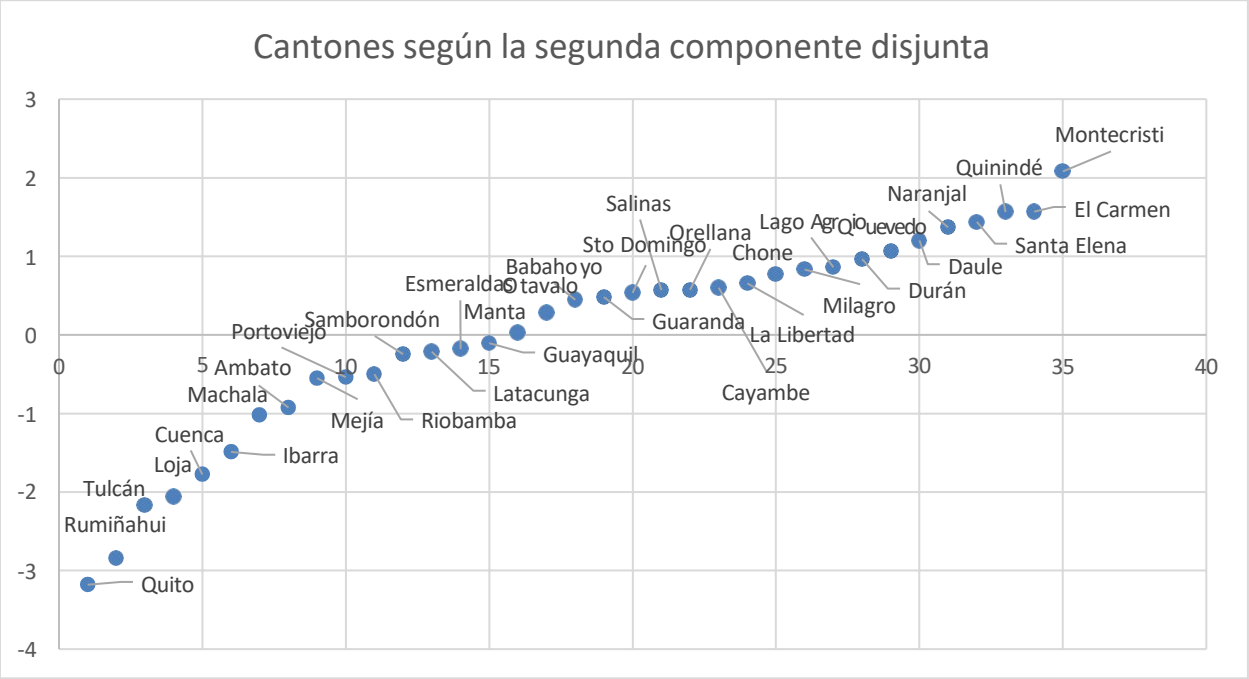

Figura 3.14: Orden ascendente de los cantones según la segunda componente disjunta

Así, Quito es uno de los cantones que tiene la tasa de contagio más alta, pero a la vez su nivel de pobreza es el menor que los cantones restantes. Montecristi es el cantón en el sentido opuesto, es decir, el cantón con la tasa de contagios más baja, y el nivel de pobreza más alto. El cantón Guayaquil, al que pertenece la segunda ciudad más poblada del país está en una posición intermedia respecto a este eje contagios-pobreza.

El estudio comprueba que los cantones más afectados son aquellos a los que pertenecen las tres ciudades más grandes tanto en población como en recursos. Esto es, los cantones Quito. Guayaquil y Cuenca. El comportamiento de la pandemia en las regiones naturales en que se divide Ecuador (Costa, Sierra y Oriente) es bastante similar. Es decir, se tiene un conglomerado en torno al origen con uno o dos cantones alejados de esta concentración central (los de mayor población y recursos). Las correlaciones entre variables tal como se interpretan en el HJ Biplot clásico no se pueden aplicar al Disjoint HJ Biplot, debido a que la proyección obtenida deforma la posición de los ejes disjuntos. En este análisis sólo hay dos opciones: que los ejes sean colineales o que sean ortogonales.

Capítulo 4

ANÁLISIS DE COMPONENTES PRINCIPALES FUNCIONALES DISJUNTAS

Los últimos avances en las diferentes tecnologías de computación, trasmisión y almacenamiento de información han permitido monitorear procesos en tiempo real obteniéndose datos que son funciones o datos funcionales. Así como los datos multivariantes se encuentran en el espacio vectorial $\mathbb{R}^n$, los datos funcionales son elementos de un espacio de Hilbert de dimensión infinita. Por esta razón se hace necesario disponer de resultados básicos sobre este tipo de espacios. En este capítulo se expone una breve introducción a la teoría fundamental de dichos espacios.

El capítulo está distribuido de la siguiente manera:

4.1 Introducción.

4.2 Fundamentos teóricos

4.3 Análisis de Componentes Principales Funcionales Clásico

4.4 Propuesta de cálculo de componentes principales funcionales disjuntas.

4.5 Aplicación del Análisis de Componentes Principales Funcionales Disjuntas.

4.1 Introducción

Si bien los datos funcionales son de naturaleza infinito dimensional son medidos de forma discreta en determinado intervalo finito de la recta real. La frecuencia a la que se realizan las mediciones ha aumentado en forma notable gracias a los últimos avances tecnológicos, lo que hace que cada dato funcional tenga una alta dimensión, Estas mediciones discretas gracias a técnicas de interpolación y aproximación permiten reconstruir la función en cuestión (J. O. Ramsay & Silverman, 2005). Aún si consideramos la versión discreta de una función se tiene que el número de individuos (curvas) es menor que el número de variables (mediciones). Presentándose la llamada “maldición de la dimensionalidad”. Esto va en contra del supuesto del Análisis Multivariante en que el número de objetos o individuos debe ser menor al número de variables.

Ejemplos de datos funcionales los podemos encontrar en varios campos del quehacer científico como espectrometría. Mediciones de condiciones climáticas como temperatura, humedad presión atmosférica, contaminación y polución atmosférica (Di Salvo et al., 2015). Ecología (Embling et al., 2012). Genética (Leng & Müller, 2006). Cotizaciones de la bolsa (Müller et al., 2011). Curvas de consumo de electricidad. Datos de tipo médico, como potenciales eléctricos como los que nos proporcionan electrocardiogramas o electroencefalogramas (Viviani et al., 2005), (Sørensen et al., 2013). En (Hadjipantelis & Müller, 2018) se detallan diversas aplicaciones al denominado "Big Data" La estadística funcional se ha mostrado como una potente herramienta para el análisis de procesos estocásticos que se desarrollan en el tiempo y el espacio.

El parámetro de las funciones consideradas como datos funcionales es por lo general el tiempo, sin embargo también puede representar distancias (estadística espacial) o frecuencias (como en espectrometría) (Pourshoghi et al., 2016).

Una de las primeras técnicas multivariantes que se aplicó en el contexto funcional fue el PCA, al que también denominaremos PCA clásico. Esta técnica busca una reducción de la dimensión del espacio en que yace un conjunto de datos a través de la obtención de un nuevo sistema ortonormal de referencia, mediante un procedimiento de maximización de la inercia de la nube de datos disponible.

En el PCA los individuos son representados por vectores en un espacio finito dimensional y el conjunto de datos por una matriz. Por ello el PCA clásico está íntimamente ligado a la teoría de matrices y operadores en espacios de dimensión finita. En su contraparte funcional tenemos el Análisis de Componentes Principales Funcionales (FPCA por sus siglas en inglés) en el cual los individuos son representados por funciones, esto es, por objetos de dimensión infinita. Por esta razón el FPCA está relacionado con el Análisis Funcional y la teoría de operadores sobre espacios de dimensión infinita (J. O. Ramsay & Silverman, 2005), (Kokoska & Reimherr, 2017).

A diferencia del Análisis Multivariante en que la unidad de información es un punto (vector) en determinado espacio vectorial $\mathbb{R}^p$ en el Análisis de Datos Funcionales (ADF)

la unidad de información es una función en el espacio vectorial de las funciones cuadrado integrables (Lu, 2007). Esto implica un cambio en la metodología de análisis de los datos funcionales. Necesitamos todo el andamiaje matemático disponible y la geometría que nos proporcionan los espacios de Hilbert separables. Es la estructura de estos espacios la que permite una interpretación de las componentes principales funcionales acorde con la utilizada en el PCA.

Básicamente se supone que las variables aleatorias funcionales son procesos estocásticos, que pertenecen a un espacio de Hilbert infinito dimensional. Mientras que los datos funcionales se pueden considerar como realizaciones (o trayectorias) de dicho proceso estocástico (Valderrama et al., 2000). Estas realizaciones son las funciones observadas (son sus mediciones los datos de que disponemos), cuyo argumento toma valores en un intervalo de la recta real y por tanto son vectores en un espacio de dimensión infinita.

¿Cómo se logra transferir un espacio de dimensión infinita a un entorno computacional que sólo admite representaciones finitas? Aquí tenemos que hacer un supuesto: que las trayectorias del proceso estocástico se pueden aproximar tanto como se dese por una base finita de funciones generadoras.

Una de las ventajas del enfoque funcional es el de obtener funciones como componentes principales que son combinaciones lineales de una base ortonormal de funciones continuas y por ende también son continuas. Esto posibilita calcular sus derivadas, es decir velocidades o tasas de cambios y por supuesto también la aceleración del proceso representado (J. Ramsay & Ramsey, 2002).

Tanto en el ambiente multivariante como en el funcional las componentes principales se presentan como combinaciones lineales de vectores, vectores en $\mathbb{R}^p$ en el primer caso y funciones en el segundo, que forman una base ortonormal finita. Presentándose el problema de interpretación de las componentes principales planteado en el capítulo 1. ¿Cuáles son las funciones que realmente pesan en determinada componente principal? ¿Cuáles elegir? ¿Cuáles descartar? En respuesta a estas interrogantes se presenta la

metodología de las componentes principales disjuntas calculadas mediante la optimización de enjambre de partículas adaptada al contexto funcional.

De la revisión bibliográfica realizada no existen antecedentes de la aplicación de las componentes disjuntas al caso funcional. Más aún de la aplicación de la optimización por enjambre de partículas para el cálculo de dichas componentes.

Los espacios de Hilbert extienden las propiedades de espacios de dimensión finita con producto interno tal como $\mathbb{R}^p$, de esta manera se hace posible generalizar los resultados tanto del álgebra como del análisis a espacios abstractos que pueden ser de dimensión infinita, como es el caso del espacio de las variables aleatorias funcionales. La completitud de los espacios de Hilbert garantiza que el límite de cualquier sucesión de sus elementos pertenezca al mismo espacio, posibilitando de esta forma la representación de un operador en término de sus valores y vectores propios.

En el contexto multivariante un vector en $\mathbb{R}^d$ es una sucesión finita de *d* componentes. El número *d* es la dimensión del espacio $\mathbb{R}^d$. En el contexto funcional las funciones, si bien son "observadas" en conjunto finito de puntos, son consideradas como objetos que pertenecen a un espacio de dimensión infinita. Las propiedades de este espacio son las que van a determinar el comportamiento e interacción de los objetos que a él pertenecen. Es deseable que la geometría del espacio vectorial finito dimensional $\mathbb{R}^d$, que resulta muy útil en la interpretación de las posiciones de los individuos proyectadas en los planos principales, se pueda extender a espacios de funciones de dimensión infinita. Por tanto, las nociones de distancia, tamaño, medida angular y ortogonalidad se puedan extender a espacios de funciones. La clave de todo esto está en definir un producto interno en el espacio funcional.

La teoría de los espacios de Hilbert es una teoría clásica, completamente terminada, sin cabos sueltos. Puesto que este trabajo está dirigido a la aplicación de determinados resultados del análisis funcional a la estructura estadística que opera con datos que son

funciones, las demostraciones de los teoremas enunciados en este capítulo y el siguiente han sido omitidas. Se las puede encontrar en cualquier texto de análisis funcional, en particular en dos excelentes libros "Theoretical Foundations of Functional Data Analysis, with an Introduction to Linear Operators" (Hsing & Eubank, 2013), a cuya estructura de presentación nos hemos ceñido, y el texto "Introduction to Functional Data Analysis" (Kokoska & Reimherr, 2017).

4.2 Fundamentos Teóricos

El PCA clásico parte de una matriz de datos **X** centrada, a partir de ella obtiene su matiz de varianzas covarianzas, esta matriz es simétrica semi definida positiva, lo que implica que sus valores propios son no negativos. La base teórica del PCA clásico radica en la descomposición espectral de la matriz de varianzas covarianzas. En el ámbito funcional se necesita un símil que permita obtener valores propios no negativos del operador de varianzas covarianzas.

En primer lugar, sección 4.2.1, se describe el espacio en el cual se va a trabajar, esto es en el espacio $L^p(T)$, el espacio de las funciones cuadrado integrables, y las propiedades de los operadores lineales sobre dichos espacios. La matriz de varianzas covarianzas pasa a convertirse en un operador lineal sobre $L^p(T)$. En el contexto de las transformaciones lineales, acotado es sinónimo de continuo. El análogo a la operación matricial $\mathbf{Av}$ para el cálculo de los valores propios es el operador lineal descrito en (4.2.6) y su núcleo viene a ser el equivalente de la matriz de varianzas covarianzas. En la sección 4.2.2 se describen los operadores auto adjuntos que son el equivalente de las matrices simétricas, tal como lo es la matriz de varianzas covarianzas. Además de ser simétricas este tipo de matrices es semi definida positiva. También se generaliza este concepto a operadores lineales sobre $L^p(T)$. En la sección 4.2.3 se definen los operadores compactos que permiten una aproximación por medio de operadores de rango finito. Es decir, los podemos aproximar desde un punto de vista matricial. Este punto es clave puesto que permite pasar de un ambiente infinito dimensional a uno de dimensión finita. Ya que el operador funcional que

se utiliza para generalizar la expresión matricial $\mathbf{Av} = \lambda \mathbf{v}$ es compacto, y el número de valores propios o bien es finito o forma una sucesión que converge a cero. En la sección 4.2.4 se establece la generalización de la descomposición espectral de una matriz a operadores lineales auto adjuntos y compactos. En la sección 4.2.5 se definen los operadores de Hilbert Schmidt en los cuales se puede generalizar la norma matricial clásica o de Frobenius. En la sección 4.2.6 tenemos a los operadores de la clase traza que extiende la idea de que la norma se puede calcular como la suma de sus valores propios. Tal como se tiene en el contexto multivariante. En la sección 4.2.7 se establece el teorema de Mercer que es el equivalente al teorema de descomposición espectral en el contexto matricial.

4.2.1 Espacios $L^p(T)$ y operadores lineales

Dentro de los espacios de Hilbert hay uno que tiene particular interés para el análisis de datos funcionales es el denominado espacio $L^2(T)$, este es el espacio al cual van a pertenecer las funciones bajo estudio.

Definición 4.1.- Sea $(T, \mathcal{B}, \mu)$ un espacio medible y sea $p \geq 1$, se define el espacio $L^p(T)$ como el conjunto de las funciones medibles en T que satisfacen:

$$\int_T |f|^p d\mu < \infty \tag{4.2.1}$$

Si $f \in L^p(T)$, se define:

$$\|f\|_p = \left(\int_T |f|^p d\mu \right)^{1/p} \tag{4.2.2}$$

Donde $T = [a,b] \subset \mathbb{R}$ es un intervalo compacto, $\mathcal{B}$ es la $\sigma-$álgebra de Borel de los subconjuntos de $[a,b]$, y μ es la medida de Lebesgue en $\mathbb{R}$, por lo que $d\mu$ se puede reemplazar por dt .

De entre todos los espacios $L^p(T)$ con $p \geq 1$, el único que es espacio de Hilbert es el espacio $L^2(T)$:

$$\int_T f^2(t)dt < \infty$$

Su producto interno está dado por:

$$\forall f_1, f_2 \in L^2(T), \quad \langle f_1, f_2 \rangle = \int_T f_1(t) f_2(t)\, dt \tag{4.2.3}$$

A los elementos de $L^2(T)$ se les denomina funciones de cuadrado integrable.

Recordemos que $L^p(T)$ es el conjunto cociente cuyos elementos son clases de equivalencia.

En el contexto multivariante se trabaja con datos definidos en espacios de dimensión finita. Así, una matriz de tamaño $m \times n$ se puede interpretar como una transformación lineal que va del espacio $\mathbb{R}^n$ en el espacio $\mathbb{R}^p$. En el caso funcional tenemos aplicaciones lineales que van de un espacio de dimensión infinita en otro espacio de dimensión infinita. Si el espacio de partida es igual al de llegada tenemos los llamados operadores lineales. En el caso de datos funcionales tenemos que el espacio vectorial es $L^2(T)$. En el análisis multivariante, las matrices son de tamaño finito y por ende representan a operadores continuos. En $L^p(T)$ los operadores acotados son continuos.

Una aplicación lineal $L: \mathbf{V}_1 \to \mathbf{V}_2$ se dice acotada si existe una constante $c \geq 0$ tal que:

$$\forall v \in \mathbf{V}_1, \ \|L(v)\|_{\mathbf{V}_2} \leq c \|v\|_{\mathbf{V}_1} \tag{4.2.4}$$

Esto implica que, si *L* es acotada, la imagen de un conjunto acotado es también un conjunto acotado.

Una aplicación lineal $L:\mathbf{V}_1 \to \mathbf{V}_2$. L es acotada, si y sólo si es continua.

Sean $\mathbf{V}_1$ y $\mathbf{V}_2$ dos espacios vectoriales. Al conjunto de todas las transformaciones lineales acotadas de $\mathbf{V}_1$ en $\mathbf{V}_2$ se le nota por $\mathcal{B}(\mathbf{V}_1,\mathbf{V}_2)$.

Con las operaciones usuales de funciones, esto es, suma y multiplicación por escalar, $\mathcal{B}(\mathbf{V}_1,\mathbf{V}_2)$ es un espacio vectorial.

En particular, si **V** es un espacio normado, al conjunto de todos los funcionales lineales acotados de **V** en $\mathbb{R}$ se denota por $\mathcal{B}(\mathbf{V},\mathbb{R})$ y se denomina espacio dual de **V**.

Traza de un operador lineal:

Sea una aplicación lineal $L:\mathbb{R}^n \to \mathbb{R}^n$, $B=\{e_i\}$ la base canónica de $\mathbb{R}^n$, y sea **A** su matriz asociada en la base *B*. Se tiene que la traza de *L* está dada por:

$$\operatorname{tr}(L)=\operatorname{tr}(\mathbf{A})=\sum_{i=1}^{n} a_{ii}=\sum_{i=1}^{n}\left\langle A^{\dagger}e_i, e_i\right\rangle=\sum_{i=1}^{n}\left\langle Ae_i, e_i\right\rangle$$

En analogía se define la traza de $\mathcal{F}\in\mathcal{B}(\mathbf{V}_1,\mathbf{V}_2)$ por:

$$\operatorname{tr}(\mathcal{F})=\sum_{i=1}^{\infty}\left\langle \mathcal{F}(e_i), e_i\right\rangle \tag{4.2.5}$$

Definición 4.2.- Consideremos el espacio $L^2(T)$. Sea *K* una función cuadrado integrable sobre $T\times T$. Sea $f\in L^2(T)$, se define el operador lineal $\mathcal{F}$ por:

$$\left\lceil\left\lfloor \mathcal{F}(f)\right\rceil\right\rfloor(\cdot)=\int_T K(\cdot,u)f(u)du \tag{4.2.6}$$

Denominado operador integral. *K* se llama núcleo del operador $\mathcal{F}$.

El operador $\mathcal{F}$ es acotado y por tanto continuo.

4.2.2 Operadores Auto-adjuntos

Sean H_1 y H_2 dos espacios de Hilbert, con productos internos $\langle\cdot|\cdot\rangle_1$ y $\langle\cdot|\cdot\rangle_2$ respectivamente. Para cada elemento $\mathcal{F}$ de $\mathcal{B}(H_1, H_2)$ existe un único elemento $\mathcal{F}^*$ de $\mathcal{B}(H_2, H_1)$ tal que:

$$\forall x_1 \in H_1 \, \forall x_2 \in H_2, \ \langle \mathcal{F}(x_1), x_2 \rangle_2 = \langle x_1, \mathcal{F}^*(x_2) \rangle_1 \tag{4.2.7}$$

El operador $\mathcal{F}^*$ se denomina operador adjunto de $\mathcal{F}$. Si H_1=H_2 y si $\mathcal{F}^* = \mathcal{F}$, $\mathcal{F}$ se denomina operador auto-adjunto. En el contexto matricial, el concepto de operador adjunto corresponde al de matriz transpuesta, y el de operador auto-adjunto corresponde al de matriz simétrica.

Sea $\mathcal{F} \in \mathcal{B}(H_1, H_2)$, se tiene que:

1. $(\mathcal{F}^*)^* = \mathcal{F}$
2. $\|\mathcal{F}^*\| = \|\mathcal{F}\|$
3. $\|\mathcal{F}^*\mathcal{F}\| = \|\mathcal{F}\|^2$

Definición 4.3.- Un operador auto-adjunto $\mathcal{F}$ sobre un espacio de Hilbert se dice que es:

1. Semidefinido positivo si satisface la condición:

$$\forall x \in H, \ \langle \mathcal{F}(x), x \rangle \geq 0 \tag{4.2.8}$$

2. Definido positivo si cumple que:

$$\forall x \in H, \ \langle \mathcal{F}(x), x \rangle > 0 \tag{4.2.9}$$

Sea un operador $\mathcal{F} \in \mathcal{B}(H)$, Si $\mathcal{F}$ es semidefinido positivo, entonces existe un único operador $\mathcal{S} \in \mathcal{B}(H)$ tal que:

$$\mathcal{S}^2 = \mathcal{F} \tag{4.2.10}$$

Se nota por $\mathcal{F}^{1/2}$ al operador raíz cuadrada $\mathcal{S}$.

4.2.3 Operadores compactos

Dentro de los operadores lineales nos interesa en particular, en el análisis de datos funcionales, los denominados operadores compactos pues se pueden aproximar por medio de operadores de rango finito cuyas propiedades son similares a las de los operadores lineales de dimensión finita. Esto es, presentan propiedades análogas a las de las matrices de varianza covarianza, usadas en el contexto multivariante.

Definición 4.4.- Sean dos espacios vectoriales normados X_1 y X_2. Una aplicación lineal $\mathcal{F}: X_1 \to X_2$ se dice que es compacta si transforma conjuntos acotados en conjuntos relativamente compactos (en conjuntos cuya clausura es compacta).

Una consecuencia inmediata de la definición anterior que toda aplicación lineal compacta es continua.

Si $\mathcal{F}$ es compacta, entonces $\overline{\mathrm{Im}(\mathcal{F})}$es separable.

Una trasformación lineal $\mathcal{F}$ de rango finito tiene por imagen a un subespacio de dimensión finita y por tanto es isomorfa a algún espacio $\mathbb{R}^p$ con $p \in \mathbb{N}$.

Sea aplicación lineal $\mathcal{F}: X_1 \to X_2$ de rango finito, entonces $\mathcal{F}$ es una aplicación compacta.

Teorema 4.1.- Sean H_1 y H_2 dos espacios de Hilbert. Sea un operador compacto $\mathcal{F}: H_1 \to H_2$, entonces $\mathcal{F}$ es el límite de una sucesión de operadores de rango finito $\mathcal{F}_n: H_1 \to H_2$ con $n \in \mathbb{N}$. Es decir, $\lim_{n\to\infty} \left\| \mathcal{F} - \mathcal{F}_n \right\| = 0$.

El teorema 6.11 nos dice que $\mathcal{F}$ se puede aproximar tanto como se requiera por una aplicación lineal de rango finito.

4.2.4 Teorema de descomposición espectral

En el ámbito multivariante, el núcleo fundamental del análisis de componentes principales es la descomposición espectral de la matriz de varianzas-covarianzas que es semi-definida positiva. En el contexto funcional, en analogía con el caso multivariante, también es de suma importancia calcular los valores y vectores propios de los llamados operadores de covarianza, que se detallarán más adelante.

Definición 4.5.- Sea H un espacio de Hilbert. Se nota por $\mathcal{B}(H)$ al espacio vectorial de los operadores lineales acotados sobre H.

Definición 4.6.- Sea $\mathcal{F} \in \mathcal{B}(H)$, Se dice que $\lambda \in \mathbb{R}$ es un valor propio de $\mathcal{F}$ si existe un vector $e \in H$, $e \neq \mathbf{0}$ tal que:

$$\mathcal{F}(e) = \lambda e \tag{4.2.11}$$

e se denomina vector propio asociado al valor propio λ.

Si H es un espacio funcional e se denomina función propia asociada a λ.

Definición 4.7.- Sea λ un valor propio de $\mathcal{F} \in \mathcal{B}(H)$, se define el espacio propio asociado a λ por $E_{\lambda} = Ker\ \mathcal{F} - \lambda I\ =\ v \in H\ /\ \mathcal{F}(v) = \lambda v$

Teorema 4.2.- Sea $\mathcal{F} \in \mathcal{B}(H)$ y sean los valores propios de λ_j, $j = 1,2,\ldots$
Supongamos que los valores propios de $\mathcal{F}$ son todos distintos y diferentes de cero, con vectores propios asociados e_j, $j = 1,2,\ldots$ respectivamente. Entonces:

1. Los vectores propios e_j, $j = 1,2,\ldots$ son linealmente independientes.
2. Si $\mathcal{F}$ es auto-adjunto, los vectores propios e_j son mutuamente ortogonales.

Teorema 4.3.- Sea $\mathcal{F} \in \mathcal{B}(H)$ un operador compacto.

1. $\dim E_{\lambda}$ es finita para $\lambda \neq 0$
2. El conjunto de valores propios de $\mathcal{F}$ diferentes de cero es contable.

Teorema 4.4.- Sea $\mathcal{F} \in \mathcal{B}(H)$ un operador compacto auto-adjunto, entonces

1. El conjunto de valores propios de $\mathcal{F}$ o bien es finito o bien constituye una sucesión que converge a cero.
2. Cada valor propio diferente de cero tiene multiplicidad algebraica finita.
3. Los vectores propios asociados a valores propios diferentes son ortogonales.

Si un valor propio tiene multiplicidad geométrica mayor a uno, los vectores propios correspondientes se deben calcular por medio del proceso de ortonormalización de Gram-Schmidt y así obtener una base ortonormal para el espacio propio correspondiente.

A continuación, tenemos el denominado teorema de descomposición espectral:

Teorema 4.5.- Sea $\mathcal{F} \in \mathcal{B}(H)$ un operador compacto auto-adjunto, entonces existe un conjunto de valores propios $\lambda_1, \lambda_2, \lambda_3, \ldots$ del operador $\mathcal{F}$, con sus respectivos vectores propios $e_1, e_2, e_3, \ldots$, tal que:

1. λ_j es una sucesión decreciente y si es infinita converge a cero.
2. e_j es una base ortonormal de $\overline{\mathrm{Im}(\mathcal{F})}$.
3. $\mathcal{F}$ admite la representación numerable:

$$\mathcal{F} = \sum_{j=1}^{\infty} \lambda_j e_j \otimes e_j \tag{4.2.12}$$

Por tanto, de (4.2.12) se concluye:

$$\forall x \in H,\ \mathcal{F}(x) = \sum_{i=1}^{\infty} \lambda_i \langle x, e_i \rangle e_i \tag{4.2.13}$$

Definición 4.8.- La igualdad (4.2.13) se denomina descomposición o representación espectral del operador $\mathcal{F}$.

Esta representación espectral de $\mathcal{F}$ es única en el siguiente sentido:

Si P_n es la proyección ortogonal sobre el correspondiente espacio propio E_λ, entonces la descomposición $\mathcal{F} = \sum_{j\geq 1} \lambda_j P_n$ es única.

Una consecuencia del teorema de descomposición espectral es:

$$\langle \mathcal{F}(x), x \rangle = \sum_{j=1}^{\infty} \lambda_j \langle x, e_j \rangle^2 \qquad (4.2.14)$$

De donde: $\langle \mathcal{F}(e_j), e_j \rangle = \lambda \|e_j\|^2$, $j = 1, 2, .$

Además, los valores propios de un operador semidefinido positivo no son negativos.

4.2.5 Operadores de Hilbert-Schmidt

Dentro de la clase de las transformaciones compactas hay una subclase que tiene especial interés en el análisis de datos funcionales, son los denominados operadores de Hilbert-Schmidt. En esta clase de operadores se puede generalizar la norma clásica matricial.

Definición 4.9.- Sean $\mathcal{F} \in \mathcal{B}(H_1, H_2)$. Sea e_j una base ortonormal cualquiera de H_1. Si se cumple que:

$$\sum_{j=1}^{\infty} \left\| \mathcal{F}(e_j) \right\|_2^2 < \infty$$

Se dice que $\mathcal{F}$ es una aplicación de Hilbert-Schmidt. Al conjunto de todas las transformaciones de Hilbert-Schmidt de H_1 en H_2 se lo nota por $\mathcal{B}_{HS}(H_1, H_2)$.

$\mathcal{B}_{HS}(H_1, H_2)$ es un espacio vectorial respecto a las operaciones usuales de funciones.

Toda aplicación de Hilbert-Schmidt es compacta.

Definición 4.10.- Sea una base ortonormal e_j de H_1. Se define la función $\langle \cdot | \cdot \rangle_{HS} : \mathcal{B}_{HS}(H_1, H_2) \times \mathcal{B}_{HS}(H_1, H_2) \to \mathbb{R}$ por:

$$\langle \mathcal{F}_1 | \mathcal{F}_2 \rangle_{HS} = \sum_{j=1}^{\infty} \langle \mathcal{F}_1(e_j), \mathcal{F}_2(e_j) \rangle_2 \qquad (4.2.15)$$

Teorema 4.6.- La función dada en (4.2.15) es un producto interno en $\mathcal{B}_{HS}(H_1, H_2)$.

Sea una base ortonormal e_j de H_1. Se define la norma de Hilbert-Schmidt en $\mathcal{B}_{HS}(H_1, H_2)$ por:

$$\|\mathcal{F}\|_{HS} = \left(\sum_{j=1}^{\infty} \|\mathcal{F}(e_j)\|_2^2 \right)^{1/2} \tag{4.2.16}$$

El valor de $\|\mathcal{F}\|_{HS}$ es independiente de elección de la base e_j . Si utilizamos la base ortonormal compuesta por los vectores propios de $\mathcal{F}\mathcal{F}(e_j) = \lambda_j e_j$, por lo que (4.2.16) se puede escribir de la forma:

$$\|\mathcal{F}\|_{H} = \left(\sum_{j=1}^{\infty} \lambda_i^2 \right)^{1/2} \tag{4.2.17}$$

Si $\mathcal{F} \in \mathcal{B}_{HS}(H_1, H_2)$ y $\mathcal{F}^*$ es su operador adjunto, la norma de Hilbert-Schmidt también se expresa por:

$$\|\mathcal{F}\|_{HS} = \left(\mathrm{tr}(\mathcal{F}^* \mathcal{F}) \right)^{1/2} \tag{4.2.18}$$

Donde la traza está dada por (4.2.5).

Si $H_1 = \mathbb{R}^m$ y $H_2 = \mathbb{R}^n$, la norma de Hilbert-Schmidt viene a ser la norma de Frobenius. La aplicación $\mathcal{F}$ está representada por su matriz asociada $\mathbf{A} = \lfloor a_{ij} \rceil_{n \times m}$ en las bases canónicas de $\mathbb{R}^m$ y $\mathbb{R}^n$:

Sea e_j el *j*-ésimo elemento de la base canónica de $\mathbb{R}^m$, entonces $\mathbf{A}e_j$ es igual a la *j*-ésima columna de $(a_{1j}, a_{2j}, \ldots, a_{mj})^t$. Por tanto:

$$\|\mathcal{F}(e_i)\|_2^2 = \|\mathbf{A}e_j\|_{\mathbb{R}^n}^2 = \|(a_{1j}, a_{2j}, \ldots, a_{mj})^t\|_{\mathbb{R}^n}^2$$
$$= a_{1j}^2 + a_{2j}^2 + \cdots + a_{mj}^2$$

$$\|\mathcal{F}\|_{HS} = \|A\|_{HS} = \left(\sum_{j=1} \left(a_{1j}^2 + a_{2j}^2 + \ldots + a_{mj}^2 \right) \right)^{1/2}$$
$$= \left(\sum_{j=1}^{n} \sum_{i=1}^{m} a_i^2 \right)^{1/2}$$
$$= \left(\mathrm{tr} \left(\mathbf{A}^{\dagger} \mathbf{A} \right) \right)^{1/2}$$

El espacio vectorial $\mathcal{B}_{HS}(H_1, H_2)$ con el producto interno dado en la ecuación (4.2.15) constituye un espacio de Hilbert separable.

4.2.6 Operadores de la clase traza

Sean H_1 y H_2 dos espacios de Hilbert separables. Se dice que $\mathcal{F} \in \mathcal{B}(H_1, H_2)$ es una aplicación de la clase traza si para alguna base ortonormal e_j de H_1 la cantidad:

$$\|\mathcal{F}\|_{TR} = \sum_{j=1}^{\infty} \left\langle \left(\mathcal{F}^* \mathcal{F} \right)^{1/2} e_j, e \right\rangle_1 \tag{4.2.19}$$

Es finita. En tal caso se dice que $\|\mathcal{F}\|_T$ es la norma traza de $\mathcal{F}$. El valor de la norma traza no depende de la elección de la base ortonormal. El operador de covarianza en el análisis de datos funcionales es un operador de la clase traza.

Para un operador $\mathcal{F}$ de la clase traza se cumple que $\|\mathcal{F}\|_{TR} = \left\| (\mathcal{F}^* \mathcal{F})^{1/4} \right\|_{HS}^2$. Aquí se debe entender que $(\mathcal{F}^* \mathcal{F})^{1/4} = \left((\mathcal{F}^* \mathcal{F})^{1/2} \right)^{1/2}$.

También podemos expresar:

$$\mathcal{F}^* \mathcal{F} = \left((\mathcal{F}^* \mathcal{F})^{1/4} \right)^4$$

Si $\mathcal{F}$ es un operador de la clase traza, se tiene que $\mathcal{F}$ es compacto. Si además es

es auto-adjunto con valores propios λ_j se tiene que:

$$\|\mathcal{F}\|_{TR} = \sum_{j=1}^{\infty} |\lambda_j| \tag{4.2.20}$$

$\sum_{j=1}^{\infty} \lambda_j < \infty$ se denomina la traza del operador $\mathcal{F}$, notada por $\text{tr}(\mathcal{F})$. Cuando $\mathcal{F}$ es semidefinido positivo se tiene:

$$\text{tr}(\mathcal{F}) = \|\mathcal{F}\|_{TR} = \sum_{j=1}^{\infty} \lambda_j \tag{4.2.21}$$

Que es el equivalente al resultado del análisis mutivariante para la matriz de varianzas-covarianzas.

4.2.7 Operadores integrales

En esta sección nos vamos a centrar en describir las propiedades de los llamados operadores integrales, como es el caso del operador de covarianza en el análisis de datos funcionales.

Consideramos un espacio medible (T, B, μ), donde T es un intervalo compacto de la recta real, μ es una medida finita con soporte T, y B es sigma álgebra de los borelianos de T.

Sea K una función medible en el espacio producto $T \times T$ tal que:

$$\iint_{T \times T} K^2(s,t)\, d\mu(s)\, d\mu(t) < \infty$$

Lo que indica que es de cuadrado integrable en $T \times T$.

Sea $f \in L^2(T)$, se define el operador integral $\mathcal{K}$ por:

$$[\mathcal{K}(f)](\cdot) = \int_T K(s,\cdot) f(s) d\mu(s) \tag{4.2.22}$$

La función K se denomina núcleo del operador.

$[\mathcal{K}(f)](\cdot)$ es medible y además $\mathcal{K} \in \mathcal{B}\left(L^2(T)\right)$.

Se tiene que el operador integral $\mathcal{K}$ es compacto.

Teorema 4.7.- El operador integral $\mathcal{K}$ es semidefinido positivo si y sólo si su núcleo K es semidefinido positivo.

Se tiene el siguiente teorema denominado **teorema de Mercer**, uno de los dos puntales básicos del análisis de componentes principales funcionales[1] y del llamado "machine learning" pues juega un papel importante en el análisis de la varianza de las variables aleatorias funcionales y nos permite realizar la representación en serie de un núcleo de covarianza.

Teorema 4.8.- Sea un núcleo K continuo, simétrico[2] y semidefinido positivo, y sea $\mathcal{K}$ su correspondiente operador integral. Sean $\lambda_1, \lambda_2, \lambda_3, \ldots$ valores propios del operador integral $\mathcal{K}$ con sus respectivos vectores propios $e_1, e_2, e_3, \ldots$, entonces el núcleo K admite la representación:

$$\forall s \in T \ \forall t \in T, \ K(s,t) = \sum_{j=1}^{\infty} \lambda_j e_j(s) e_j(t) \tag{4.2.23}$$

La sumatoria en (4.2.23) converge absoluta y uniformemente.

[1] El otro soporte es la representación de Karhunen-Loève, que lo trataremos en la sección 4.3.2.

[2] Simétrico en el contexto de funciones bivariadas significa que $K(s,t) = K(t,s)$

En el próximo capítulo vamos a utilizar los resultados enunciados para dar una estructura matemática al desarrollo de la teoría de las componentes principales funcionales.

4.3 Análisis de Componentes Principales Funcionales Clásico

El desarrollo de los medios tecnológicos y la conectividad a gran escala han permitido que la toma de datos se pueda realizar de forma masiva, inmediata y constante. Así, por medio de sensores, que son fácilmente asequibles y sencillos de manejar, es factible el monitoreo en tiempo real y de manera continua de diferentes eventos que anteriormente se los hacía de forma discreta y esporádica. Por ejemplo, en las estaciones meteorológicas había un termómetro de máxima y mínima, cuyas lecturas se hacían una vez por día. Ahora, un sensor nos envía la información sobre la temperatura registrada en la estación meteorológica ya sea de manera continua o a intervalos de tiempo bien pequeños.

Por ende, se abre un nuevo arquetipo en el manejo de datos estadísticos: hoy por hoy nos enfrentamos a datos que ya no son puntos en un espacio multivariante sino son funciones que pueden tener como argumento una o más variables. Dicho de otra forma, necesitamos de una estadística que maneje al dato funcional de la forma como se hacía con el dato multivariante.

La estadística funcional tiene como elemento fundamental al dato funcional que es aquel que resulta de procesos registrados de forma continua en el tiempo o en el espacio. Los datos funcionales se pueden modelizar mediante la teoría de Proceso Estocásticos. Así, una variable aleatoria funcional viene a ser representada por un proceso estocástico estacionario y las observaciones de dicha variable viene a ser realizaciones del proceso estocástico. Par describir la estructura estadística en la que se fundamenta el análisis de componentes principales funcionales en primer lugar vamos a reseñar los principales resultados que conciernen a los procesos estocásticos estacionarios.

4.3.2 Estructura estocástica del Análisis de Componentes Principales Funcionales Clásico

Definición 4.11.- Sea un espacio de probabilidad $(\Omega,\mathcal{F},P)$. Sea $T\subseteq\mathbb{R}$. Un proceso estocástico X se define de la forma:

$$\begin{aligned} X: T\times\Omega &\to S\subseteq\mathbb{R} \\ (t,\omega) &\to X(t,\omega) \end{aligned} \tag{4.3.1}$$

Donde:

T se denomina conjunto de parámetros o índices, por lo general representa al tiempo, es un intervalo de $\mathbb{R}$ de la forma $T=[0,\infty)$ o $T=[a,b]$. En análisis de datos funcionales se toma $T=[a,b]$. Sin pérdida de generalidad vamos a considerar en adelante que t representa al tiempo.

S es la imagen de la aplicación X y se denomina espacio de estados del proceso. La variable aleatoria $X(t,\omega)$ también se representa por $X_t(\omega)$. Para un valor dado de $t\in T$ $X_t(\omega)$ es una variable aleatoria, y para un $\omega\in\Omega$ dado le corresponde la función $X_t(\omega)$ que es una función que depende del tiempo denominada "realización o trayectoria del proceso". Por comodidad se suele notar a $X(t,\omega)$ por $X(t)$, sin olvidar que también es función de ω. Así, el proceso estocástico es una familia de variables aleatorias:

$$X=\{X(t,\omega)/t\in T\}=\{X(t)/t\in T\} \tag{4.3.2}$$

Definición 4.12.- Sea la función $\langle\cdot,\cdot\rangle:(\Omega,\mathcal{F},P)\times(\Omega,\mathcal{F},P)\to\mathbb{R}$ dada por:

$$\begin{aligned} \langle X,Y\rangle_\Omega &= \mathrm{E}[XY] \\ &= \int_\Omega X(w)Y(w)dP(w) \end{aligned} \tag{4.3.3}$$

Nótese que si las variables aleatorias X y Y son centradas:

1. $\langle X,Y\rangle_{\Omega}$ es su covarianza.
2. La varianza de X viene a ser:

$$\mathrm{Var}[X] = \mathrm{E}[X]^2 = \int_{\Omega} X^2(w)dP(w) = \langle X, X\rangle_{\Omega}$$

Por tanto, la desviación estándar σ_X es la norma de X,

3. El coeficiente de correlación entre X y Y está dado por :

$$\cos(\theta) = \frac{\langle X,Y\rangle_{\Omega}}{\|X\|_{\Omega}\|Y\|_{\Omega}} = \frac{\mathrm{E}[XY}{\sigma_X \quad Y} = \rho$$

El espacio $(\Omega, \mathcal{F}, P)$ con el producto interno dado por (4.3.3) y que satisfacen $\mathrm{E}[X]^2 < \infty$ es un espacio de Hilbert separable notado por $L^2(\Omega)$. Puesto que $\mathrm{E}[X]^2 > \mathrm{E}^2[X]$, $L^2(\Omega)$ es el espacio de las variables aleatorias de varianza finita.

Definición 4.13.- Si todas las variables aleatorias $X(t,\omega)$ pertenecen al espacio de Hilbert $L^2(\Omega)$ se dice que el proceso estocástico X es de segundo orden.

Definición 4.14.- Una variable aleatoria funcional es un proceso estocástico de segundo orden $X = \{X(t) / t \in T\}$ y un dato funcional $X(t)$ es una realización o trayectoria de dicho proceso estocástico *X*.

Definición 4.15.- Se dice que un proceso estocástico $X = \{X(t) / t \in T\}$ es continuo en media cuadrática si satisface:

$$\forall t \in T,\ \lim_{h\to 0} E\left[\left(X(t+h) - X(t)\right)^2\right] = 0$$

Por tanto, en promedio la distancia cuadrática de la trayectoria $X(t)$ entre dos instantes de tiempo bastante cercanos tiende a cero.

Sea $T = [a,b] \subset \mathbb{R}$, las trayectorias o realizaciones $X(t)$ del proceso X son funciones de *t* definidas en el compacto $[a,b]$. Si asumimos que las trayectorias del proceso estocástico *X* pertenecen al espacio de Hilbert $L^2(T)$ el producto interno dado en (4.2.3) nos provee de una geometría útil para describir e interpretar a los datos funcionales.

En adelante todas las variables aleatorias funcionales consideradas serán procesos estocásticos de segundo orden, continuos en media cuadrática, y sus realizaciones $X(t)$, como funciones de *t*, serán elementos de $L^2(T)$.

Definición 4.16.- Sea la variable aleatoria funcional $X = \{X(t) / t \in T\}$, se definen:

1. La función media:

$$\begin{aligned} \mu &: T \to \mathbb{R} \\ t &\to \mu(t) = \mathrm{E}[X(t)] \end{aligned} \tag{4.3.4}$$

2. La función covarianza:

$$\begin{aligned} \mathrm{C} &: T \times T \to \mathbb{R} \\ (t,s) &\to \mathrm{C}(t,s) = \mathrm{E}\lceil\lfloor (X(t) - \mu(t))(X(s) - \mu(s))\rceil\rfloor \end{aligned} \tag{4.3.5}$$

Definición 4.17.- Si $\mu(t) = 0$ para todo $t \in T$, se dice que la variable aleatoria funcional es centrada.

Si *X* no es centrada, la variable aleatoria funcional $X = \{X(t) - \mu(t) / t \in T\}$ es centrada.

Sin perder generalidad, en adelante vamos a considerar que las variables aleatorias funcionales son centradas.

Propiedades de la función covarianza $C(t,s)$:

1. Es auto-adjunta
2. Es semidefinida positiva

Teorema 4.9.- Un variable aleatoria funcional $X=\{X(t)/t\in T\}$ es continua en media cuadrática si y sólo si su función covarianza C es continua en $T\times T$.

Definición 4.18.- Sea una variable aleatoria funcional $X=\{X(t)/t\in T\}$ se define su operador covarianza $\mathcal{K}:L^2(T)\to L^2(T)$ por:

$$[\mathcal{K}(X)](t)=\int_T C(t,s)X(s)ds \tag{4.3.6}$$

El núcleo de este operador es la función covarianza C.

Como la variable aleatoria funcional es continua en media cuadrática, la función de covarianza C es continua y por tanto el operador $\mathcal{K}$ es acotado.

Propiedades del operador covarianza:

1. $\mathcal{K}$ es compacto sobre $L^2(T)$, lo que implica que se puede aproximar por un operador de rango finito tanto como se requiera.
2. Es auto-adjunto puesto que:

$$\forall X_1, X_2\in L^2(T),\ \langle \mathcal{K}(X_1), X_2\rangle = \langle X_1, \mathcal{K}(X_2)\rangle$$

3. Es semidefinido positivo:

$$\forall X_1\in L^2(T),\ \langle \mathcal{K}(X_1), X_1\rangle \geq 0$$

De la forma en que se ha definido el operador covarianza lo convierte en un operador integral de la clase traza y un operador compacto de Hilbert-Schmidt.

Sean $\{\lambda_j, e_j\}_{j=1}^{\infty}$ los valores y vectores propios del operador covarianza $\mathcal{K}$, se tiene:

$$\lceil \lfloor \mathcal{K}(e_j) \rceil \rfloor(t) = \int_T C(t,s)e_j(s)ds = \lambda_j e_j(t) \tag{4.3.7}$$

La ecuación (4.3.7) se denomina ecuación integral de Fredholm y en su resolución está la clave del análisis de componentes principales funcionales. Nótese la semejanza con su contraparte multivariante: Para determinar las componentes principales multivariantes hay que resolver un problema de maximización de varianza que al aplicar los multiplicadores de Lagrange se llega a que la solución es la misma que del problema de valores y vectores propios $\mathbf{V}\mathbf{u}_1 = \lambda \mathbf{u}_1$ [3].

En el caso funcional la matriz de varianza covarianza $\mathbf{V}$ está representada por $C(t,s)$ y el vector $\mathbf{u}_1$ por la función e_1. El producto matricial $\mathbf{V}\mathbf{u}_1$, al encontrarnos en espacio de Hilbert está dado por $\int_T C(t,s)e_j(s)ds$. En el caso matricial el número de valores propios, contando su multiplicidad es finito, en el caso funcional es infinito.

En general, la resolución de (4.3.7) es un problema mal condicionado, no hay garantías de que sus soluciones sean estables. Por esta razón, en este documento vamos a proponer una estrategia de estimación de la solución del problema, como se detallará en la subsección 4.3.4.

Puesto que el operador covarianza es semi definido positivo los valores propios de $\mathcal{K}$ son no negativos. Por el teorema de Mercer (4.2.23) se tiene que:

$$\forall s \in T \ \forall t \in T, \ C(s,t) = \sum_{j=1}^{\infty} \lambda_j e_j(s) e_j(t) \tag{4.3.8}$$

Supongamos que se desea determinar una función $\varphi_1 \in L^2(T)$ de norma unitaria tal que $\langle \mathcal{K}(\varphi_1), \varphi_1 \rangle$ sea máximo. Puesto que $\mathcal{K}$ es semidefinido positivo todos sus valores propios λ_j son no negativos. Ordenando los λ_j en orden decreciente tenemos $\lambda_1 \geq \lambda_2 \geq \cdots$

[3] Ver Capítulo 1.

Vamos a suponer que $\lambda_i \neq \lambda_j$ para todo $i \neq j$ lo que asegura que exista que por cuanto sus funciones propias son mutuamente ortogonales.

Al igual que se hizo en el caso multivariante tenemos que maximizar:

$$\left\langle \mathcal{K}(\varphi_1),\varphi_1 \right\rangle = \left\langle \sum_{j=1}^{\infty} \lambda_j \left\langle \varphi_1, e_j \right\rangle, \varphi_1 \right\rangle = \sum_{j=1}^{\infty} \lambda_j \left\langle \varphi_1, e_j \right\rangle^2 \tag{4.3.9}$$

Con la restricción $\sum_{j=1}^{\infty} \lambda_j \left\langle \varphi_1, e_j \right\rangle = 1$

Puesto que el mayor valor propio es λ_1 con función propia asociada e_1, tomamos $\varphi_1 = e_1$ o $\varphi_1 = -e_1$:

$$\left\langle \mathcal{K}(e_1), e_1 \right\rangle = \left\langle \sum_{j=1}^{\infty} \lambda_j \left\langle e_1, e_j \right\rangle, e_1 \right\rangle = \sum_{j=1}^{\infty} \lambda_j \left\langle e_1, e_j \right\rangle^2 = \lambda_1 \tag{4.3.10}$$

Y el valor máximo es λ_1, al igual que en el caso multivariante (como se puede apreciar en el capítulo I).

Una vez determinada la función principal φ_1, deseamos encontrar una función principal φ_2 tal que maximice $\left\langle \mathcal{K}(\varphi_2),\varphi_2 \right\rangle$ y que sea ortogonal a φ_1, siguiendo el mismo proceso antes descrito vamos a comprobar que $\varphi_2 = e_2$. Así sucesivamente si deseamos encontrar una función principal φ_3 que maximice $\left\langle \mathcal{K}(\varphi_3),\varphi_3 \right\rangle$ y que sea ortogonal a φ_2 y φ_3.

Para determinar las componentes principales funcionales el enfoque utilizado es similar a su contraparte multivariante, es decir, expresar a las funciones observadas como una combinación lineal generalizada de la forma:

$$\alpha_1 = \int_T X(t) f(t) dt$$

Donde $f \in L^2(T)$.

En (Valderrama et al., 2000) se muestra que la varianza de α está dada por:

$$\mathrm{Var}[\alpha_1] = \mathrm{E}\left[\left(\int_T X(t) f(t) dt\right)^2\right] = \mathrm{E}\left[\left(\int_T X(t) f(t) dt\right)\left(\int_T X(s) f(s) ds\right)\right]$$

De donde:

$$\mathrm{Var}[\alpha] = \mathrm{E}\left[\iint_{T\times T} X(t) X(s) f(t) f(s) dt ds\right]$$

Ingresando la esperanza a la integral:

$$\mathrm{Var}[\alpha_1] = \iint_{T\times T} \underbrace{\mathrm{E}[X(t)X(s)]}_{C(t,s)} f(t)\, f(s)\, dt ds$$

$$\mathrm{Var}[\alpha_1] = \iint_{T\times T} C(t,s) f(t) f(s)\, dt ds$$

$$\mathrm{Var}[\alpha_1] = \int_T \mathcal{K}(f(t)) f(t) dt = \langle \mathcal{K}(f(t)), f(t) \rangle$$

De (4.3.9) y (4.3.10) el máximo para $\mathrm{Var}[\alpha_1] = \lambda_1$ el mayor valor propio del operador covarianza, y f es la función principal asociada a λ_1, esto es $f = e_1$. Y así con las demás combinaciones lineales:

$$\alpha_j = \int_T X(t) e_j(t) dt \tag{4.3.11}$$

Las variables α_j se denominan scores, mientras que las funciones e_j son las componentes principales de la variable aleatoria funcional X.

La covarianza entre dos scores $\alpha_i = \int_T X(t) e_i(t) dt$ y $\alpha_j = \int_T X(t) e_j(t) dt$ con $i \neq j$, está dada por:

$$\mathrm{Cov}[\alpha_i, \alpha_j] = \mathrm{E}[\alpha_i, \alpha_j] = \mathrm{E}\left[\left(\int_T X(t) e_i(t) dt\right)\left(\int_T X(s) e_j(s) ds\right)\right]$$

De donde:

$$\mathrm{Cov}\left[\alpha_i,\alpha_j\right]=\mathrm{E}\left[\iint_{T\times T} X(t)X(s)e_i(t)e_j(s)dtds\right]$$

$$\mathrm{Cov}\left[\alpha_i,\alpha_j\right]=\iint_{T\times T}\underbrace{\mathrm{E}\left[X(t)X(s)\right]}_{C(t,s)}e_i(t)e_j(s)dtds$$

$$\begin{aligned}\mathrm{Cov}\left[\alpha_i,\alpha_j\right]&=\int_T \mathcal{K}(e_j(s))e_i(t)dtds\\&=\left\langle \mathcal{K}(e_j),e_i(t)\right\rangle\end{aligned}$$

Aplicando (4.2.14)

$$\mathrm{Cov}\left[\alpha_i,\alpha_j\right]=\sum_{j=1}^{\infty}\lambda_j\left\langle e_i,e_j\right\rangle^2$$

Como las funciones principales son ortogonales: $\left\langle e_i,e_j\right\rangle=0$. Por tanto, $\mathrm{Cov}\left[\alpha_i,\alpha_j\right]=0$

A continuación, tenemos la denominada representación de Karhunen-Loève, que permite expresar a un proceso estocástico en forma de serie al estilo Fourier, pero con la diferencia que los coeficientes α_j son variables aleatorias.

Teorema 4.10.- Sea una variable aleatoria funcional $\mathrm{X}=\left\{X(t)/t\in T\right\}$. Sean $\lambda_1\geq\lambda_2\geq\cdots\geq 0$ los valores propios de su operador covarianza $\mathcal{K}$, con sus respectivas funciones propias $e_1,e_2,\cdots$. Se tiene que:

$$X(t)=\sum_{j=1}^{\infty}\alpha_j e_j(t) \tag{4.3.12}$$

La convergencia de la sumatoria en (4.3.12) es en el sentido de la norma de $L^2(T)$.

La representación proporcionada por el teorema de Kahunen-Loève permite, al igual que en el caso multivariante, una reducción de la dimensión considerando sólo los primeros *k* sumandos de la representación (4.3.12) para un valor de *k* lo suficientemente grande. Otra forma de reducir la dimensionalidad es proyectar los datos funcionales en un espacio

generado por una base finita de funciones como series de Fourier, trazadores cúbicos o núcleos.

4.3.3 Valores propios y funciones principales.

En el análisis multivariante disponemos de una matriz de datos X de tamaño $n \times p$, cuyas entradas x_{ij} representan a los valores observados de la variable X_j ($j = 1,.,p$) en el individuo $i = i,.,n$. En el contexto funcional se dispone de una variable aleatoria funcional $X = \{X(t) / t \in T\}$, y de una muestra de *n* datos funcionales $X_1(t), \ldots, X_n(t)$ observados en los puntos $t_1, t_2, \ldots, t_p \in T$, que suponemos están igualmente espaciados. Estos datos son convertidos en elementos pertenecientes a espacios funcionales generados por bases finitas. Este es otro paso clave en FPCA, pues permite pasar de un contexto infinito dimensional a uno finito dimensional, es decir a un espacio en el que se puede usar representaciones matriciales.

Esta conversión se realiza a través de métodos de interpolación o de suavizado que se desarrollan en la próxima subsección.

Estimación puntual de la función media:

Definición 4.19.- Se define el estimador muestral de la media por:

$$\bar{X}(t) = \frac{1}{n} \sum_{j=1}^{n} X_j(t) \tag{4.3.13}$$

Teorema 4.11.- El estimador $\bar{X}(t)$ es insesgado, esto es $\mathrm{E}\lceil \bar{X}(t) \rceil = \mathrm{E}[X(t)]$ y converge en media cuadrática a $\mathrm{E}[X(t)]$.

Estimación de la función covarianza:

Definición 4.20.- Se define el estimador muestral de la covarianza por:

$$\overline{C}(s,t) = \frac{1}{n}\sum_{j=1}^{n}\left(X_j(s)-\bar{X}(s)\right)\left(X_j(t)-\bar{X}(t)\right) \tag{4.3.14}$$

Teorema 4.12.- El estimador $\overline{C}(s,t)$ es insesgado, es decir $\mathrm{E}\left[\overline{C}(s,t)\right] = \mathrm{E}\left[C(s,t)\right]$ y converge en media cuadrática a $\mathrm{E}\left[C(s,t)\right]$.

4.3.4 Estimación de las componentes principales funcionales.

Con el estimador muestral $\overline{C}$ del núcleo de covarianza C, podemos construir un estimador del operador de covarianza $\mathcal{V}$ y resolver la ecuación integral de Fredholm resultante:

$$\left[\mathcal{V}(e_j)\right](t) = \int_T \overline{C}(t,s)e_j(s)ds = \lambda_j e_j \tag{4.3.15}$$

Los valores y funciones propias soluciones de (4.3.15) son estimadores de sus correspondientes que son soluciones de la ecuación (4.3.7). Por comodidad en la notación los vamos a llamar de la misma manera.

Para estimar la solución de (4.3.15) vamos a utilizar el método descrito en (J. O. Ramsay & Silverman, 2005) y (Hsing & Eubank, 2013), que consiste en representar a la mencionada ecuación en forma matricial, para ello a cada función X_j se escribe como combinación lineal de una base de funciones $B = \{\psi_1, \psi_2, ., \psi_K\}$:

$$X_j(t) = \sum_{k=1}^{K} c_{jk}\psi_k \tag{4.3.16}$$

La elección de K depende de muchos factores, el más importante es si los datos son interpolados o ajustados. Si se tiene la segunda opción el valor de K debe ser mucho menor que n. ¿Qué tan menor debe ser K? Depende de la bondad de ajuste que resulte del valor de K elegido, del número de puntos n en que se midieron las variables X_j, y de la base elegida.

En formato matricial:

$$\mathbf{X} \quad \mathbf{C} \tag{4.3.17}$$

Donde:

$$\mathbf{X} \quad \begin{bmatrix} X_1(s) \\ \vdots \\ X_n(s) \end{bmatrix}_{n\times 1},$$

$$\mathbf{C} \quad \begin{bmatrix} c_{11} & \cdots & c_{1K} \\ \vdots & \ddots & \vdots \\ c_{nk} & \cdots & c_{nK} \end{bmatrix}_{n\times K},$$

$$\begin{bmatrix} \psi_1(t) \\ \vdots \\ \psi_K(t) \end{bmatrix}_{K\times 1}$$

Con esta notación se tiene que el núcleo de covarianza es:

$$\overline{C(s,t)} \quad {}^{\dagger}(s)\,\frac{\mathbf{C}^{\dagger}\mathbf{C}}{n} \quad (t) \tag{4.3.18}$$

Sea $\mathbf{G}$ la matriz de Gram de la base B:

$$\mathbf{G} \quad \left[\left\langle \psi_i, \psi_j \right\rangle\right] \quad \left[\int_T \psi_i(t)\psi_j(t)dt\right]_{K\times K}$$

Puesto que estamos trabajando en el campo de los números reales, $\mathbf{G}$ es definida positiva.

Si la base *B* es ortonormal, $\mathbf{G}$ es la matriz identidad.

Supongamos que una función propia e_j del operador de covarianza $\mathcal{K}$, asociado al valor propio λ, tiene la representación:

$$e_j(t) \quad \sum_{k\ 1}^{K} b_k \psi_k(t)$$

En formato matricial:

$$e_j(s) \qquad ^{\dagger}(s)\mathbf{b}(s)$$

Reemplazando en (4.3.15):

$$\int_T \overline{C}(t,s)\, e_j(s)ds \quad \int_T \quad ^{\dagger}(s)\, \frac{\mathbf{C}^{\dagger}\mathbf{C}}{n} \quad (t) \quad ^{\dagger}(t)\mathbf{b}(t)\, dt$$

De donde: $\int_T \overline{C}(t,s)\, e_j(s)ds \qquad ^{\dagger}(s)\, \frac{\mathbf{C}^{\dagger}\mathbf{C}}{n} \underbrace{\int_T \quad (t) \quad ^{\dagger}(t)\mathbf{b}(t)\, dt}_{\mathbf{G}} \mathbf{b}(t)$

$$\int_T \overline{C}(t,s)\, e_j(s)ds \qquad ^{\dagger}(s)\, \frac{\mathbf{C}^{\dagger}\mathbf{C}}{n}\mathbf{G}\mathbf{b}(t)$$

De donde:

$$^{\dagger}(s)\, \frac{\mathbf{C}^{\dagger}\mathbf{C}}{n}\mathbf{G}\mathbf{b}(t) \quad \lambda e_j(s) \quad \lambda \ ^{\dagger}(s)\, \mathbf{b}(s)$$

Lo que implica:

$$\frac{\mathbf{C}^{\dagger}\mathbf{C}}{n}\mathbf{G}\mathbf{b} \quad \lambda\mathbf{b}$$

La matriz $\frac{\mathbf{C}^{\dagger}\mathbf{C}}{n}\mathbf{G}$ no es simétrica. Dado que $\mathbf{G}$ es definida positiva existe una única matriz $\mathbf{G}^{1/2}$ tal que $\mathbf{G}^{1/2}\mathbf{G}^{1/2} = \mathbf{G}$. Consideremos el cambio de variable $\mathbf{b} = \mathbf{G}^{-1/2}\mathbf{u}$. Es decir, $\mathbf{u} = \mathbf{G}^{1/2}\mathbf{b}$:

$$\frac{\mathbf{C}^{\dagger}\mathbf{C}}{n}\mathbf{G}^{1/2}\mathbf{G}^{1/2}\mathbf{b} = \lambda\mathbf{G}^{-1/2}\mathbf{G}^{1/2}\mathbf{b}$$
$$\frac{\mathbf{C}^{\dagger}\mathbf{C}}{n}\mathbf{G}^{1/2}\mathbf{u} = \lambda\mathbf{G}^{-1/2}\mathbf{u}$$

Multiplicando a ambos lados por $\mathbf{G}^{1/2}$:

$$\mathbf{G}^{1/2}\frac{\mathbf{C}^{\dagger}\mathbf{C}}{n}\mathbf{G}^{1/2}\mathbf{u} = \lambda\mathbf{G}^{1/2}\mathbf{G}^{-1/2}\mathbf{u} = \lambda\mathbf{u}$$

Lo que implica que $\mathbf{u}$ es el vector propio de la matriz simétrica $\mathbf{G}^{1/2}\frac{\mathbf{C}^{\dagger}\mathbf{C}}{n}\mathbf{G}^{1/2}$ asociado al valor propio λ. Si calculamos λ y $\mathbf{u}$, obtenemos $\mathbf{b}$.

Por tanto, el análisis de componentes principales funcionales es equivalente al análisis de componentes principales multivariante de la matriz $\mathbf{C}\mathbf{G}^{1/2}$. Además, si la base *B* es ortonormal $\mathbf{G} = \mathbf{I}$ es equivalente al análisis de componentes principales multivariante de la matriz $\mathbf{C}$.

En efecto, la matriz de varianzas covarianzas de $\mathbf{C}\mathbf{G}^{1/2}$ es:

$$\frac{1}{n}\left(\mathbf{C}\mathbf{G}^{1/2}\right)^{\dagger}\left(\mathbf{C}\mathbf{G}^{1/2}\right) = \frac{1}{n}\left(\mathbf{G}^{1/2}\right)^{\dagger}\mathbf{C}^{\dagger}\mathbf{C}\mathbf{G}^{1/2} = \mathbf{G}^{1/2}\frac{\mathbf{C}^{\dagger}\mathbf{C}}{n}\mathbf{G}^{1/2}$$

Y si la base *B* es ortonormal, como en el caso de las bases de Fourier, la matriz de varianzas covarianzas está dada por:

$$\frac{\mathbf{C}^{\dagger}\mathbf{C}}{n} \tag{4.3.19}$$

El número máximo de componentes principales es la cardinalidad de la base *B*, es decir *K*. Si se aplica algún método de suavización a los datos originales se recomienda que el número de componentes principales funcionales sea mucho menor que *K*. No existe un criterio teórico para determinar el valor *K*, sino que depende en gran medida del tipo de problema que se esté analizando.

Como se manifestó anteriormente los datos funcionales son generados de forma discreta ¿Cómo convertir estos datos discretos en funciones? Para ello necesitamos expresarlos como combinación lineal de alguna base conocida. En la próxima sección haremos un repaso de las bases funcionales más utilizadas.

4.3.5 Bases para representar a los datos funcionales

El análisis de datos funcionales los datos son tomados de manera discreta en la forma de pares ordenados (t_i, y_i) donde y_i es el valor que toma la variable aleatoria *X* en el punto t_i . Los valores $t_0,t_1, \ldots,t_p$ constituyen una partición del intervalo T $[a,b]$ de tal forma que a $t_0 < t_1 < \cdots < t_p$ b. En este trabajo vamos a suponer que la partición es regular: $t_{i-1} - t_i$ $\frac{b-a}{p}$ para todo i $1,2,\ldots,p$.

Para convertir los datos discretos (t_i, y_i) con i $1,2,\ldots,p$ en una función y_i $x(t_i)$ hay que tener en cuenta si las medidas contienen error o no. Si no hay error en las observaciones o éste es muy pequeño podemos optar por interpolar los datos discretos: y_i $x(t_i)$.

Por el contrario, si el error de medición no es despreciable, y_i $x(t_i) + \varepsilon_i$, hay que pensar en un ajuste de los datos mediante algún procedimiento estadístico para eliminar el error ε_i

En cualquiera de los casos descritos se intenta representar a los datos funcionales como una combinación lineal de unas pocas funciones que tengan propiedades que permitan su fácil tratamiento matemático, como por ejemplo que sean elementos de un espacio de Hilbert. En esta subsección vamos a concretarnos al supuesto de que los datos fueron observados sin error de medición.

Sea una base funcional $B \quad \{\psi_1, \psi_2, \ldots, \psi_K\} \subseteq L^2(T)$ deseamos construir una función *x* tal que sea combinación lineal de los elementos de la base *B*:

$$x(t) \quad \sum_{k\ 1}^{K} c_k \psi_k(t)$$

En notación matricial:

$$\mathbf{x} \quad \mathbf{c}^{\dagger} \tag{4.3.20}$$

Donde $\mathbf{c} \quad [c_1, c_2, \ldots, c_K]^{\dagger}$ y $\quad [\psi_1(t), \psi_2(t), \ldots, \psi_K(t)]^{\dagger}$

$\mathbf{c}$ contiene las coordenadas de la función *x* en la base *B*.

La calidad, en el sentido de la bondad de ajuste, de la función *x* depende tanto de la base *B* seleccionada, como del valor *K* elegido.

Entre las bases más utilizadas en análisis de datos funcionales tenemos las bases de Fourier, las bases de splines, y los núcleos. Un criterio para seleccionar una base es considerar el tipo de datos disponibles: si existe cierta periodicidad se suele utilizar las bases de Fourier, caso contrario se prefiere las bases de trazadores cúbicos o splines.

Bases de Fourier.

Si los datos funcionales exhiben rasgos de periodicidad en el intervalo T, la mejor elección es optar por las bases de Fourier. En este caso las funciones $\psi_k \in B$ se definen de la siguiente manera:

$$\psi_0(t) \quad \frac{1}{\sqrt{T}}, \ \psi_{2k-1}(t) \quad \frac{\text{sen}(k\omega t)}{\sqrt{T/2}}, \ \psi_{2k}(t) \quad \frac{\cos(k\omega t)}{\sqrt{T/2}} \quad \text{para } k \ \ 1,2,\ldots,K-1$$

Es decir:

$$x(t) \simeq c_0 + c_1 \frac{\text{sen}(\omega t)}{\sqrt{T/2}} + c_2 \frac{\cos(\omega t)}{\sqrt{T/2}} + c_3 \frac{\text{sen}(2\omega t)}{\sqrt{T/2}} + c_4 \frac{\cos(2\omega t)}{\sqrt{T/2}} + \cdots$$

Donde $c_0 \quad \psi_0(t) \quad \frac{1}{\sqrt{T}}$, ω es la frecuencia y define el período $P \quad \frac{2\pi}{\omega}$. Si la partición de T es regular y su longitud es igual al período P, la base B es ortonormal. En la figura 4.1 se exhiben los 5 primeros elementos de la base de Fourier. En este ejemplo $T \quad [0,12]$ y por tanto $c_0 \quad \frac{1}{\sqrt{12}} \quad 0.288$ que corresponde a la línea recta. La curva (b) corresponde a ψ_1, la (c) a ψ_2, la (d) a ψ_3, y la curva (e) a ψ_4.

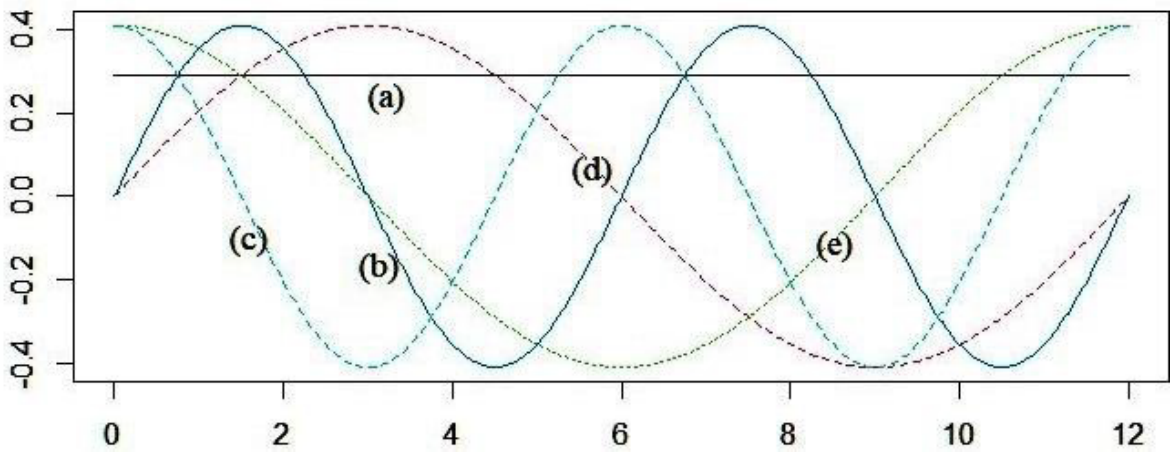

Figura 4.1: Base de Fourier

Bases de trazadores cúbicos

Un spline es una curva formada por funciones polinómicas y racionales definidos por tramos, unidos entre sí de acuerdo a determinadas condiciones de regularidad, es decir, continuidad. Consideremos la partición regular $P \ \{t_0, t_1, \ldots, t_p\}$ del intervalo $T \ [a,b]$. Los $p+1$puntos t_i se denominan nodos.

Sea S una función definida sobre el intervalo T, se dice que es una función spline de grado *k* si:

1. En cada tramo $\lceil \lfloor t_{i-1}, t_i \rfloor$ S es una función polinomial de grado menor o igual a *k.*
2. *S* tiene su derivada de orden $(k-1)$ continua en *T.*
3. $S^{(k-1)}(t_0) \quad S^{(k-1)}(t \ ) \quad 0$

Si, además, en cada nodo $\{t_0, t_1, \ldots, t_p\}$ se satisface $S(t_i) \quad y_i$ se tiene un spline interpolador.

Cualquier combinación lineal de splines definidas sobre un mismo intervalo *T* y una misma partición *P* de *T*, también es una función spline.

A manera de ejemplo un interpolador spline de grado 1 se define de la siguiente manera:

$$S(t) \quad \begin{cases} S_0(t) \quad \alpha_0 t + \beta_0 & t \in \lfloor t_0, t_1 \rceil \\ S_1(t) \quad \alpha_1 t + \beta_1 & t \in \lfloor t_1, t_2 \rceil \\ \quad \vdots & \quad \vdots \\ S_{p-1}(t) \quad \alpha_{p-1} t + \beta_{p-1} & t \in \lfloor t_{p-1}, t_p \rceil \end{cases}$$

Tal que, $S_{i-1}(t_i) \quad S_i(t_i)$ para $1 \le i \le p-1$ que asegura la continuidad, y $S''(t_0) \quad S''(t_p) \quad 0$ En análisis de datos funcionales los splines más utilizados son los de grado 3, también llamados trazadores cúbicos. Un interpolador spline de grado 3 se define como:

$$S(t) \quad \begin{cases} S(t) \quad \alpha t^3 + \beta t^2 + \gamma t + \delta & t \in \lfloor t, t \rceil \\ S^0(t) \quad \alpha t^3 + \beta t^2 + \gamma t + \delta^0 & t \in \lfloor t^0, t^1 \rceil \\ \quad \vdots & \quad \vdots \\ S_{p-1}(t) \quad \alpha_{p-1} t^3 + \quad _{p-1} t^2 + \quad _{p-1} t + \delta_{p-1} & t \in \lfloor t_{p-1}, t_p \rceil \end{cases}$$

Tal que:

1. Continuidad: $S_{i-1}(t_i) \quad S_i(t_i)$ para $1 \le i \le p-1$
2. Suavidad: *S* tiene segunda derivada continua en *T*.

Condiciones de los extremos[4]: $S''(t_0) \quad S''(t_p) \quad 0$ Como las funciones S_i son polinomios cúbicos su primera y segunda derivadas son funciones continuas, razón por la que *S* es una curva suave.

Un ejemplo de interpolación mediante un spline cúbico lo podemos apreciar en la figura 4.2.

[4] La función *S* fuera del intervalo *T* está constituida por funciones lineales.

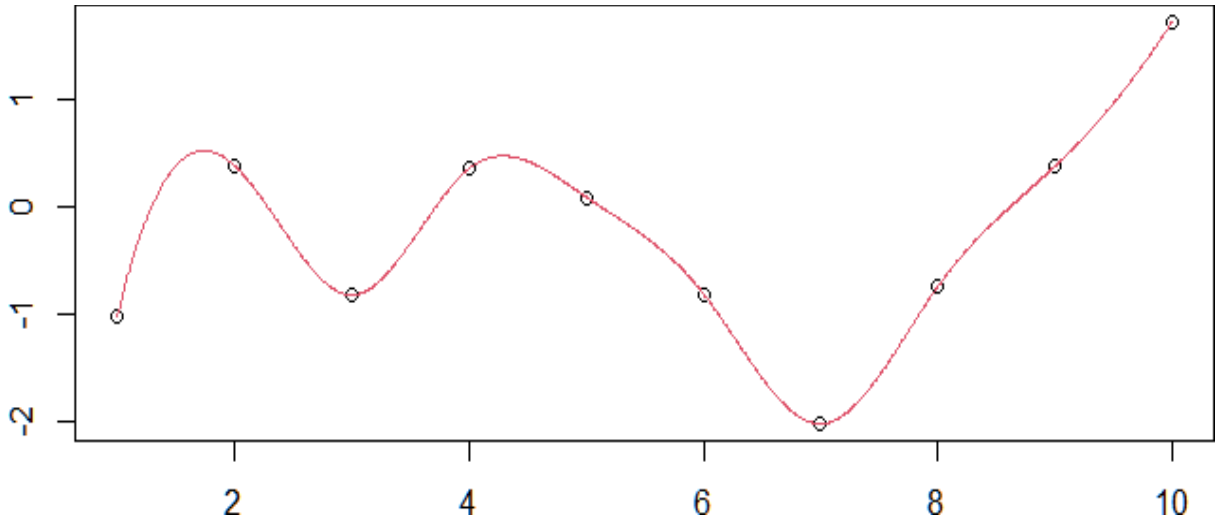

Figura 4.2: Interpolación mediante splines cúbicos

Para determinar S se tiene que calcular los coeficientes de cada S_i: $\alpha_i, \beta_i, \gamma_i, \delta_i$ para $0 \le i \le p-1$. En total $4p$ incógnitas. Se tiene que resolver un sistema lineal que satisfaga las siguientes condiciones:

1. Condiciones de interpolación: $S_i(t_i) \quad y_i$, $0 \le i \le p$, $p+1$ ecuaciones

2. Condiciones de continuidad: $S_{i-1}(t_i) \quad S_i(t_i)$, $1 \le i \le p-1$, $p-1$ ecuaciones

3. Condiciones de continuidad de la primera derivada: $S'_{i-1}(t_i) \quad S'_i(t_i)$, $1 \le i \le p-1$, $p-1$ ecuaciones

4. Condiciones de continuidad de la primera derivada: $S''_{i-1}(t_i) \quad S''_i(t_i)$, $1 \le i \le p-1$, $p-1$ ecuaciones

5. Condiciones de los extremos: $S''_0(t_0) \quad S''_{p-1}(t_p) \quad 0$, dos ecuaciones.

En total se tienen $4p$ ecuaciones. Por tanto, se tiene un sistema lineal de $4p$ ecuaciones con $4p$ incógnitas. Si el determinante asociado al sistema de ecuaciones es diferente de cero, se tiene que existe solución única.

B-splines

Las bases cuyos elementos son funciones splines se denominan bases B-splines. Las funciones que conforman una base B-spline son splines con soporte mínimo respecto a su grado, suavidad y a la partición del intervalo *T*. El soporte de una función es el conjunto sobre el cual son diferentes de cero (J. O. Ramsay & Silverman, 2005).

El primer paso para definir una función spline es generar una partición $\{\zeta_0, \zeta_1, \ldots, \zeta_\ell\}$ del intervalo T $[a,b]$ en ℓ subintervalos:

$$\lceil \zeta_0, \zeta_1), \lceil \zeta_1, \zeta_2), \ldots, \lceil \zeta_{\ell-2}, \zeta_{\ell-1}), [\zeta_{\ell-1}, \zeta_\ell]$$

Hay $\ell - 1$ puntos internos $\zeta_1, \ldots, \zeta_{\ell-1}$ que se denominan puntos de corte o de control. Si añadimos los puntos extremos a y b tenemos $\ell + 1$ puntos de control.

Sobre cada intervalo la función spline es un polinomio de orden *m*. Se entiende por orden de un polinomio como el número de coeficientes necesarios para definirlo. Por ejemplo, el polinomio de grado 2 $\alpha t^2 + \beta t + \gamma$, tiene tres coeficientes, por tanto su orden es 3. Siempre el orden del polinomio es una unidad mayor que el grado.

Si el orden de los polinomios es mayor a dos, los polinomios adyacentes se unen de manera suave en cada punto de corte, lo que entre otras cosas implica que los valores de las funciones polinómicas adyacentes deben ser iguales en el punto de corte. La condición de suavidad implica que las derivadas hasta el orden $m - 2$ también deben ser iguales en estos puntos de corte donde se unen los polinomios.

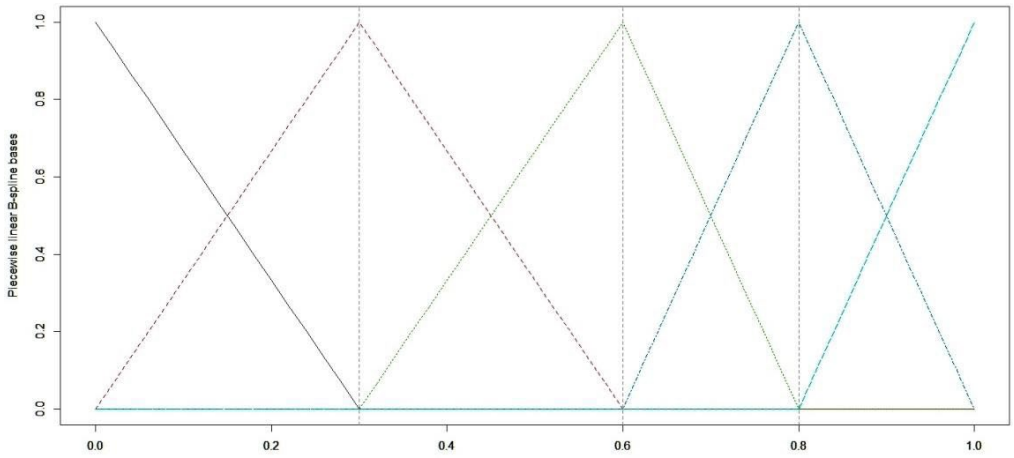

Figura 4.3: Base B-spline lineal con tres puntos de control

En el ejemplo de la figura 4.3 se aprecian cuatro splines que conforma una base B-spline. Las líneas verticales marcan la posición de los tres puntos de control, ubicados en las posiciones 0.3, 0.6 y 0.8 del eje de las abscisas. Cada elemento de la base está formado por cuatro segmentos de recta (cada spline tiene diferente color). Los segmentos de recta están determinados por dos parámetros, su intercepto y su pendiente, por lo que tienen dos grados de libertad. Así, en cada spline hay 8 grados de libertad a considerar. Hay que restar los grados de libertad que corresponde a las restricciones de continuidad, que en total son tres, puesto que se requiere la continuidad en los puntos de control. En total se tienen 5 grados de libertad.

En la figura 4.4 se aprecia una base B-spline con cuatro splines y tres puntos de control ubicados en 0.25, 0.50 y 0.75. Cada segmento de curva es una parábola, y por ende tiene tres parámetros, esto es, tres grados de libertad. En cada spline se tienen doce grados de libertad. En cada punto de control hay que tomar en cuenta dos condiciones de continuidad, para la función spline y para su derivada. Son dos grados de libertad menos por punto de control. En total, cada elemento de la base B-spline tiene 12-6=6 grados de libertad.

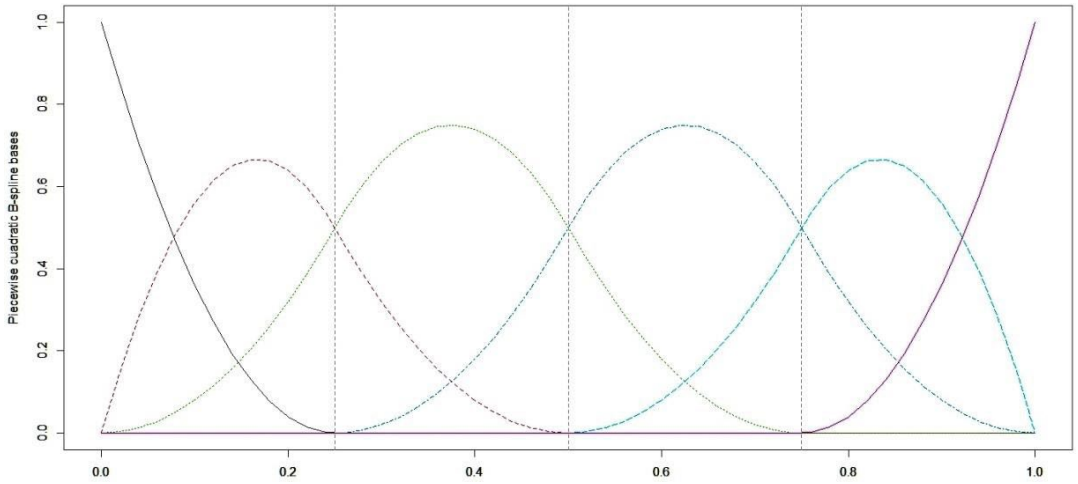

Figura 4.4: Base B-spline cuadrática con tres puntos de control

Podemos generalizar esta idea por medio de la siguiente regla: El número de grados de libertad es igual al orden del polinomio más el número de puntos de control. Si no hay puntos de control, el spline es un polinomio simple, como se aprecia en la figura 4.5.

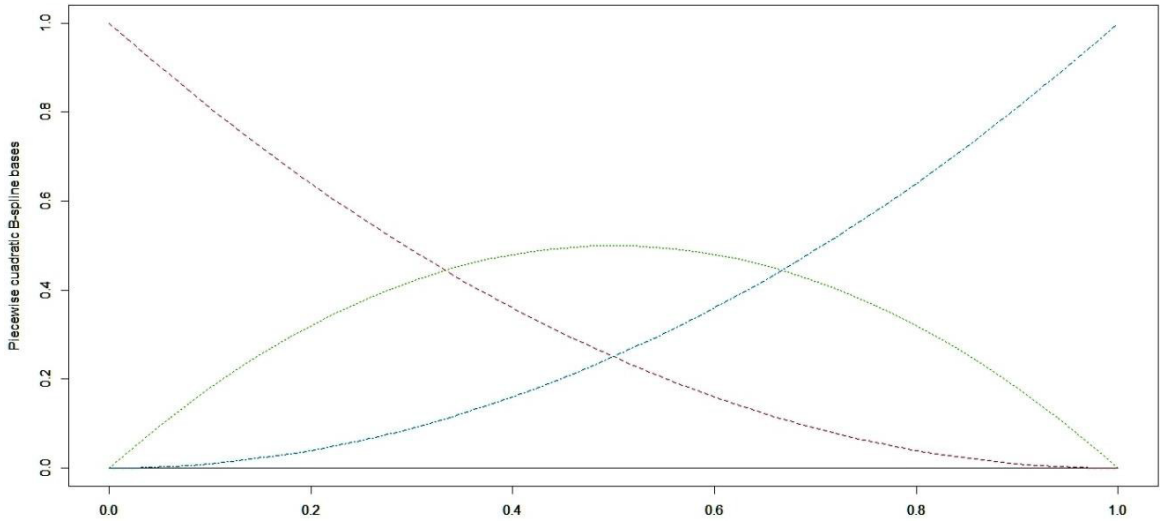

Figura 4.5: Base B-spline cuadrática sin puntos de control

Para aumentar la flexibilidad de una curva spline se tiene que incrementar el número de puntos de control. Los puntos de control no necesariamente están ubicados a la misma distancia, como se puede notar en la figura 4.4. Esto permite moverlos de tal forma que se optimice el ajuste cuando se está aproximando una función dada. Incluso pueden llegar a fusionarse. Cuando esto sucede hay una pérdida en la continuidad. De esta manera, se puede modelizar cambios abruptos en una derivada o en el valor de la función en puntos de control predeterminados.

De esta manera, podemos diseñar cambios abruptos en una derivada o incluso en un valor de función en puntos de ruptura predeterminados. El lector interesado debe consultar a De Boor (2001) para más detalles.

Las B-splines se determinan de forma numérica mediante el algoritmo recursivo de Boor:

La i-ésima función de la base B-spline de grado k $\psi_{i,k}$ se define de la siguiente manera:

$$\psi_{i,0} \begin{cases} 1 & t_i \le t < t_{i+1} \\ 0 & \text{de otra forma} \end{cases}$$

$$\psi_{i,k}(t) \quad \frac{t - t_i}{t_{i+k} - t_i}\psi_{i,k-1}(\mathrm{t}) + \frac{t_{i+k+1} - t}{t_{i+k+1} - t_{i+1}}\psi_{i+1,k-1}(t)$$

Con $i \quad 0,1,\ldots,n$ y $k \ge 1$

Las propiedades de las funciones B-splines son:

1. Cada elemento $\psi_i \in B$ es una función spline
2. Cualquier combinación de los elementos de B es una función spline.
3. Cualquier función spline definida en términos de m y de la partición $\{\zeta_0, \zeta_1, \ldots, \zeta_\ell\}$ se puede expresar como combinación lineal de las funciones de la base B-spline.

Las funciones spline están implementadas en el leguaje R en las librerías fda y fda.usc.

4.3.6 Ajuste de datos funcionales por medio de bases funcionales

Si los errores de medición de los datos funcionales no son despreciables tenemos que considerar el esquema:

$$y_i \quad x(t_i) + \varepsilon_i, \quad i \quad 1,2,\ldots,p \tag{4.3.21}$$

En notación matricial:

$$\mathbf{y} \quad \mathbf{x} + \mathbf{e} \tag{4.3.22}$$

Donde:

$$y \quad \begin{bmatrix} y_i \\ \vdots \\ y_p \end{bmatrix}, \mathbf{x} \quad \begin{bmatrix} x(t_1) \\ \vdots \\ x(t_p) \end{bmatrix} \text{ y } \mathbf{e} \quad \begin{bmatrix} \varepsilon_1 \\ \vdots \\ \varepsilon_p \end{bmatrix}$$

En esta situación tenemos que utilizar algún procedimiento de ajuste de datos para minimizar el error de medición de (4.3.21). La técnica a emplearse es la de ajuste por los mínimos cuadrados.

Para aplicar el modelo de regresión por mínimos cuadrados ordinarios los errores ε_i deben cumplir las hipótesis (Su et al., 2012):

1. $\varepsilon_i \sim N(0,\sigma_\varepsilon^2)$, con $\sigma_\varepsilon^2 < \infty$

2. Son independientes entre sí.

3. Los errores no dependen de la variable *x*.

Supongamos que disponemos de una base funcional $B \ \{\psi_1,\psi_2,\ldots,\psi_K\}$ y sea desea estimar la función *x* expresándola como combinación lineal de *B*:

$$x(t) \quad \sum_{k\ 1}^{K} c_k \psi_k(t) \tag{4.3.23}$$

Utilizando la notación (4.3.20): $\mathbf{x} \quad \mathbf{c}^{\dagger}$, el error cuadrático se define como el cuadrado de la norma $\|\mathbf{y} - \mathbf{x}\|$:

$$\begin{aligned} \mathrm{EC}(\mathbf{y}|\mathbf{x}) \quad & \|\mathbf{y} - \mathbf{x}\|^2 \\ & \sum_{i\ 1}^{p} \left(y_i - \sum_{k\ 1}^{K} c_k \psi_k(t_i) \right) \end{aligned} \tag{4.3.24}$$

Ésta es una norma en $\mathbb{R}^p$ que es un espacio de Hilbert.

Sea la matriz $\boldsymbol{\psi} \quad \left[\psi_k(t_i) \right]_{p \times K}$:

$$\begin{aligned} \mathrm{EC}(\mathbf{y}|\mathbf{x}) \quad & \|\mathbf{y} - \boldsymbol{\Psi}\mathbf{c}\|^2 \\ & \left[\mathbf{y} - \boldsymbol{\Psi}\mathbf{c}\right]^{\dagger} \left[\mathbf{y} - \boldsymbol{\Psi}\mathbf{c}\right] \end{aligned} \tag{4.3.25}$$

Derivando (4.3.25) con respecto a $\mathbf{c}$ e igualando a cero:

$$\frac{\partial \mathrm{EC}(\mathbf{y}|\mathbf{x})}{\partial \mathbf{c}} \quad 2\boldsymbol{\Psi}\boldsymbol{\Psi}^{\dagger}\mathbf{c} - 2\boldsymbol{\Psi}^{\dagger}\mathbf{y} \quad 0$$

De donde:

$$\hat{\mathbf{c}} \quad \left(\boldsymbol{\Psi}^{\dagger}\boldsymbol{\Psi}\right)^{-1} \boldsymbol{\Psi}^{\dagger}\mathbf{y}$$

Los valores estimados de las observaciones y_i se calculan por:

$$\begin{aligned} \mathbf{y} \quad & \boldsymbol{\Psi}\hat{\mathbf{c}} \\ & \underbrace{\boldsymbol{\Psi}\left(\boldsymbol{\Psi}^{\dagger}\boldsymbol{\Psi}\right)^{-1}\boldsymbol{\Psi}^{\dagger}}_{\mathbf{P}}\mathbf{y} \end{aligned}$$

La regresión lineal por mínimos cuadrados ordinarios también se puede interpretar desde un punto de vista geométrico algebraico. Si consideramos que $\boldsymbol{\Psi}$ es la matriz asociada a una transformación lineal $\mathbb{R}^K \to \mathbb{R}^p$, entonces, $\mathbf{P}$ es la matriz de proyección ortogonal sobre el espacio columna de la matriz $\mathbf{X}$. $\mathbf{P}$ es simétrica e idempotente ($\mathbf{P}^2 \quad \mathbf{P}$). Además,

$\mathbf{I}-\mathbf{P}$ es la matriz de proyección en el complemento ortogonal del espacio columna de la matriz $\mathbf{X}$, que es la imagen de la transformación lineal. La matriz $\mathbf{I}-\mathbf{P}$ también es simétrica e idempotente.

Se definen los residuos del modelo de regresión lineal $\xi_i \quad y_i - y_i$. En forma matricial:

$$\boldsymbol{\xi} \quad \mathbf{y}-\mathbf{y}$$

De donde:

$$\begin{aligned} \boldsymbol{\xi} \quad & \mathbf{y}-\mathbf{y} \\ & \mathbf{y}-\mathbf{P}\mathbf{y} \\ & (\mathbf{I}-\mathbf{P})\,\mathbf{y} \end{aligned}$$

Por tanto $\langle \boldsymbol{\xi} , \mathbf{y} \rangle \quad 0$, puesto que:

$$\begin{aligned} \langle \boldsymbol{\xi} , \mathbf{y} \rangle \quad & \boldsymbol{\xi}^{\dagger}\mathbf{y} \quad \lceil \lfloor (\mathbf{I}-\mathbf{P})\mathbf{y} \rceil \rfloor^{\dagger} (\mathbf{P}\mathbf{y}) \\ & \mathbf{y}^{\dagger}(\mathbf{I}-\mathbf{P})\mathbf{P}\mathbf{y} \\ & \mathbf{y}^{\dagger}(\mathbf{P}-\mathbf{P}\mathbf{P})\mathbf{y} \\ & 0 \end{aligned}$$

Si alguna de las hipótesis sobre la conducta de los errores no se cumple, el modelo de mínimos cuadrados ordinarios no se puede utilizar. En estos casos se utilizan los mínimos cuadrados ponderados, que consiste en minimizar:

$$\mathrm{EC}(\mathbf{y}|\mathbf{x}) \quad [\mathbf{y}-\boldsymbol{\Psi}\mathbf{c}]^{\dagger}\mathbf{W}[\mathbf{y}-\boldsymbol{\Psi}\mathbf{c}] \qquad (4.3.26)$$

Donde $\mathbf{W}$ es una matriz definida positiva, llamada matriz de pesos. tenemos que:

$$\mathbf{W} \quad \boldsymbol{\Sigma}_{\xi}^{-1}$$

Donde $\boldsymbol{\Sigma}_{\xi}$ es la matriz de varianzas-covarianzas de los residuos. Puesto que se trata de una matriz de covarianzas $\boldsymbol{\Sigma}_{\xi}$ es simétrica y por tanto $\boldsymbol{\Sigma}_{\xi}^{-1}$ también lo es.

La estimación del vector $\mathbf{c}$ por mínimos cuadrados ponderados es:

$$\hat{\mathbf{c}} \quad \left(\boldsymbol{\Psi}^{\dagger}\mathbf{W}\boldsymbol{\Psi}\right)^{-1}\boldsymbol{\Psi}^{\dagger}\mathbf{W}\mathbf{y} \qquad (4.3.27)$$

En este caso la matriz de proyección es $\mathbf{P} \quad \boldsymbol{\Psi}\left(\boldsymbol{\Psi}^{\dagger}\mathbf{W}\boldsymbol{\Psi}\right)^{-1}\boldsymbol{\Psi}^{\dagger}\mathbf{W}$ y $\mathbf{y} \quad \mathbf{P}\mathbf{y}$ y los residuos del modelo de regresión lineal ponderado están dados por $\boldsymbol{\xi} \quad (\mathbf{I}-\mathbf{P})\mathbf{y}$. En este caso:

$$\begin{aligned}\left\langle \boldsymbol{\xi}\,,\,\mathbf{y}\right\rangle \quad & \boldsymbol{\xi}^{\dagger}\mathbf{y} \quad \lceil\lfloor(\mathbf{I}-\mathbf{P})\mathbf{y}\rceil\rfloor^{\dagger}(\mathbf{P}\mathbf{y}) \\ & \mathbf{y}^{\dagger}(\mathbf{I}-\mathbf{P})\mathbf{P}\mathbf{y} \\ & \mathbf{y}^{\dagger}(\mathbf{P}-\mathbf{P}\mathbf{P})\mathbf{y} \\ & 0\end{aligned}$$

La matriz de proyección $\mathbf{P}$ se puede considerar como una aplicación lineal de alisamiento o de suavización de los datos y_i. Por tanto, a la regresión lineal se le considera un método de alisamiento lineal.

Un alisador lineal permite determinar a partir de datos $y_i \quad x(t_i)+\varepsilon_i$ con errores de medición una estimación de y_i, por medio de la combinación lineal:

$$y_i \quad x(t_i) \quad \sum_{\ell\ 1}^{p} S_i(t_\ell)y_\ell \qquad (4.3.28)$$

En términos matriciales:

$$x(\mathbf{t}) \quad \mathbf{S}\mathbf{y} \qquad (4.3.29)$$

Donde

$$x(\mathbf{t}) \quad \begin{bmatrix} x(t_1) \\ \vdots \\ x(t_p) \end{bmatrix}$$

En el caso de los mínimos cuadrados ordinarios y ponderados $\mathbf{S} \quad \mathbf{P}$ la matriz de proyección.

4.4 Propuesta de cálculo de componentes principales funcionales disjuntas.

De lo que se ha expuesto, en resumen, tenemos un sencillo resultado: el PCA funcional es equivalente al PCA multivariante de la matriz $\mathbf{CG}^{1/2}$. Las componentes principales funcionales se calculan a través de las componentes principales de la matriz $\mathbf{CG}^{1/2}$.

Además, si la base es ortonormal, como en el caso de la base de Fourier, el PCA funcional es equivalente al PCA multivariante de la matriz $\mathbf{C}$ que contiene las coordenadas de las funciones *x* respecto a la base *B*. A partir de este postulado, podemos aplicar el algoritmo CBPSO DC para calcular las componentes disjuntas de la matriz $\mathbf{C}$.

Si bien ambos procedimientos devienen en un tratamiento matricial. La diferencia radica en que, si cambiamos el orden en las columnas de la matriz de datos X en el contexto matricial, su matriz de variancias covarianzas no va a cambiar, salvo en la posición de sus componentes y por ende sus valores y vectores propios serán los mismos. En cambio en el contexto funcional si cambiamos el orden de las variables significa cambiar el orden en las series de tiempo, obteniéndose una curva diferente en cada caso y por tanto su representación en la base funcional elegida va a cambiar obteniéndose una matriz $\mathbf{C}$ diferente y por ende una matriz de covarianzas diferente tanto en sus posiciones como en sus componentes. Lo que indica que la posición de las variables (el ordenamiento temporal) es crucial en el análisis.

En el siguiente algoritmo, al que se denomina PSO DFPCA (PSO Disjoint Functional Principal Analysis) se detalla la manera de como llevar a cabo el cálculo de las componentes principales disjuntas funcionales mediante el método de optimización por enjambre de partículas:

Algoritmo PSO DFPC

Partimos de una matriz centrada de datos funcionales discretos **X**. En esta matriz las variables son los instantes temporales en los cuales se tomó la muestra.

1: Elegir una base funcional finita (base de Fourier, splines)

2: Obtener la matriz **C** y $\mathbf{G}^{1/2}$ y calcular su producto $\mathbf{CG}^{1/2}$

3: Aplicar el algoritmo CBPSO DC a $\mathbf{CG}^{1/2}$ y obtener las componentes funcionales disjuntas de la matriz **C** .

4: Mostrar los resultados

4.5 Aplicación del Análisis de Componentes Principales Funcionales Disjuntas.

Vamos a considerar un conjunto de datos funcionales que se encuentran en la librería de R "fda" denominado "CanadianWeather" que contiene los datos de temperatura y precipitaciones de 35 diferentes localidades de Canadá, desde 1960 a 1994. En esta sección se utilizan dos librerías de R: la primera es la librería "fda" (J. O. Ramsay et al., 2014), y la segunda es la librería "fda.usc" (Febrero-Bande & Oviedo de la Fuente, 2011). En la figura 4.6 se muestran las curvas de temperatura de estas localidades:

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
St. Johns	-4.65	-5.33	-2.53	1.26	5.79	10.79	15.21	15.28	11.62	7.02	2.95	-1.85
Halifax	-6.16	-6.18	-1.74	3.62	9.44	14.78	18.38	18.20	13.87	8.49	3.24	-2.99
Sydney	-5.72	-6.80	-2.94	1.85	7.50	13.14	17.49	17.64	13.31	8.27	3.53	-2.03
Yarmouth	-3.22	-3.49	0.15	4.69	9.34	13.40	16.29	16.60	13.59	9.25	4.90	-0.46
Charlottvl	-8.11	-8.26	-3.44	2.32	8.76	14.44	18.29	17.98	13.44	8.03	2.57	-4.15
Fredericton	-9.92	-8.74	-2.59	4.06	10.79	16.21	19.24	18.31	13.00	7.30	1.17	-6.59
Scheffervll	-23.71	-22.38	-15.88	-7.35	0.91	8.23	12.37	10.95	5.35	-1.45	-9.20	-19.68
Arvida	-15.36	-13.23	-5.82	2.95	10.20	16.00	18.66	17.06	11.95	5.96	-1.16	-11.04
Bagottville	-16.24	-14.30	-6.59	2.27	9.33	15.37	17.94	16.46	11.02	4.98	-2.36	-11.97
Quebec	-12.78	-11.28	-4.67	3.29	10.91	16.38	19.05	17.65	12.50	6.46	-0.54	-8.98
Sherbrooke	-11.82	-10.65	-4.04	3.93	10.79	15.49	17.97	16.69	11.96	6.38	0.00	-8.22
Montreal	-10.53	-9.06	-2.51	5.72	12.99	18.04	20.83	19.42	14.52	8.28	1.66	-6.77
Ottawa	-11.08	-9.42	-2.83	5.63	12.93	17.97	20.78	19.23	14.21	7.85	1.01	-7.51
Toronto	-6.81	-6.10	-0.79	6.16	12.40	17.50	20.53	19.49	15.14	8.90	3.22	-3.32
London	-6.75	-6.21	-0.60	6.30	12.66	17.72	20.29	19.26	15.20	9.07	3.25	-3.31
Thunderbay	-15.06	-12.66	-5.64	2.68	9.04	13.86	17.59	16.48	11.13	5.26	-2.81	-11.32
Winnipeg	-18.32	-14.94	-6.79	3.82	11.70	16.92	19.59	18.34	12.24	5.51	-4.92	-14.58
The Pas	-21.34	-17.43	-9.68	0.57	8.72	14.70	17.61	16.37	9.85	3.32	-7.75	-17.85
Churchill	-27.04	-25.61	-20.18	-10.25	-1.13	6.31	11.83	11.35	5.46	-1.61	-12.64	-22.75
Regina	-16.50	-12.84	-5.65	4.22	11.47	16.37	18.86	18.03	11.62	4.99	-5.14	-13.49
Pr. Albert	-19.76	-15.86	-8.28	2.77	10.41	15.13	17.46	16.13	9.96	3.59	-7.54	-16.85
Uranium Cty	-26.89	-22.29	-15.05	-2.48	6.91	13.28	16.20	14.47	7.29	0.44	-11.77	-22.19
Edmonton	-14.07	-10.80	-5.08	3.84	10.38	14.20	15.98	15.06	9.97	4.52	-5.66	-12.08
Calgary	-9.45	-6.29	-2.24	4.28	9.77	13.94	16.25	15.64	10.66	5.59	-2.99	-8.00
Kamloops	-4.79	-0.55	4.56	9.53	14.23	18.29	20.83	20.31	15.01	8.51	1.63	-3.17
Vancouver	2.98	4.75	6.38	8.94	12.28	15.23	17.29	17.43	14.36	10.03	5.96	3.52
Victoria	3.52	4.89	6.25	8.52	11.61	14.36	16.27	16.27	13.82	9.78	5.98	3.90
Pr. George	-9.88	-5.44	-0.52	4.86	9.62	13.23	15.33	14.72	9.94	4.72	-2.98	-8.16
Pr. Rupert	0.94	2.55	3.69	5.69	8.46	10.98	12.94	13.33	11.34	7.99	4.00	1.82
Whitehorse	-18.44	-13.21	-6.97	0.55	6.75	11.71	14.01	12.44	7.17	0.56	-9.86	-15.69
Dawson	-28.83	-23.76	-13.02	-1.07	7.77	13.54	15.53	12.60	5.93	-4.15	-17.81	-25.43
Yellowknife	-27.89	-24.70	-18.15	-6.21	5.08	13.11	16.52	14.15	6.65	-1.49	-14.66	-23.98
Iqaluit	-26.24	-27.33	-23.82	-15.07	-4.40	3.39	7.66	6.81	2.20	-4.80	-12.99	-22.41
Inuvik	-28.63	-28.33	-23.70	-13.81	-0.41	10.66	13.94	10.67	3.18	-8.16	-21.50	-26.14
Resolute	-32.37	-33.01	-31.23	-23.56	-10.98	-0.58	4.12	1.81	-4.98	-15.05	-24.45	-29.13

Tabla 4.1: Datos de temperatura promedio mensual en 35 localidades de Canadá

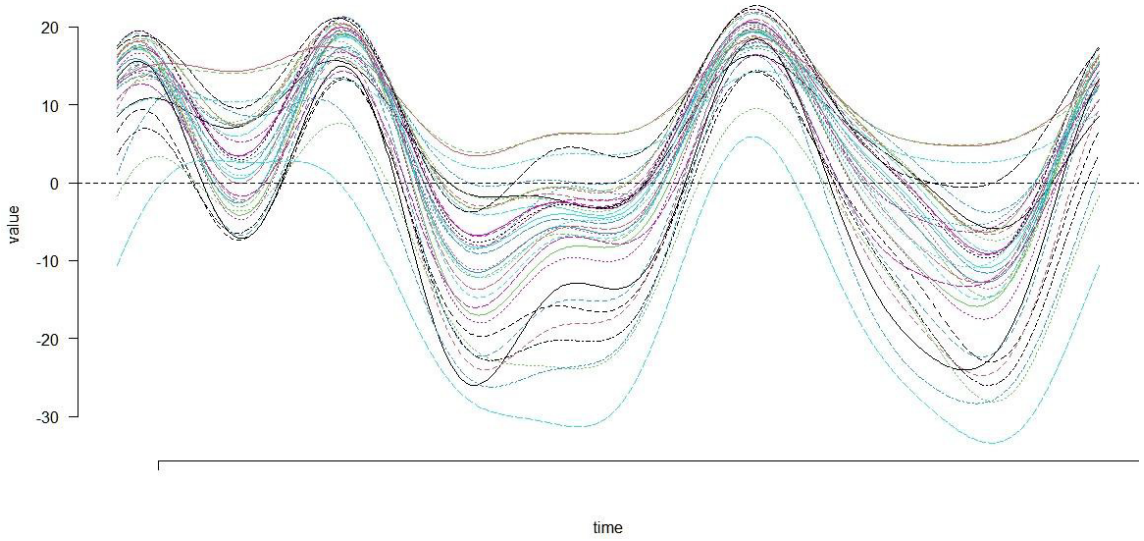

Figura 4.6: Curvas de temperatura promedio mensual en 35 localidades de Canadá

A continuación, se grafican algunas de las curvas de temperatura de forma individual:

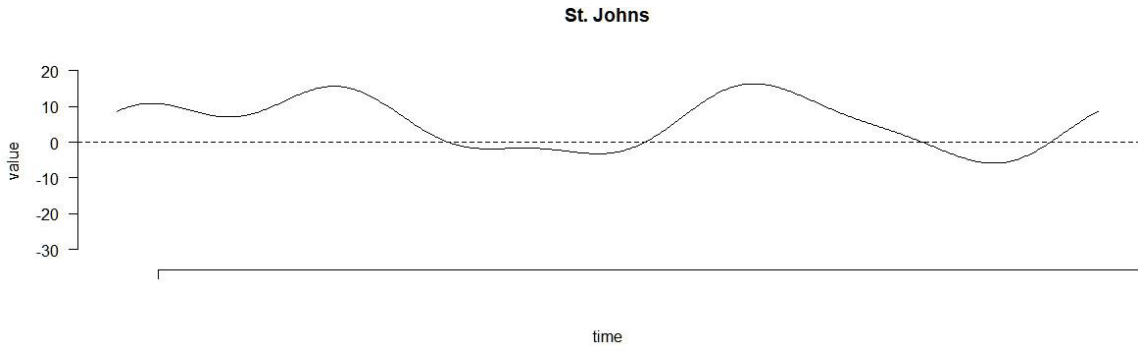

Figura 4.7: Temperaturas de la localidad de St. Johns

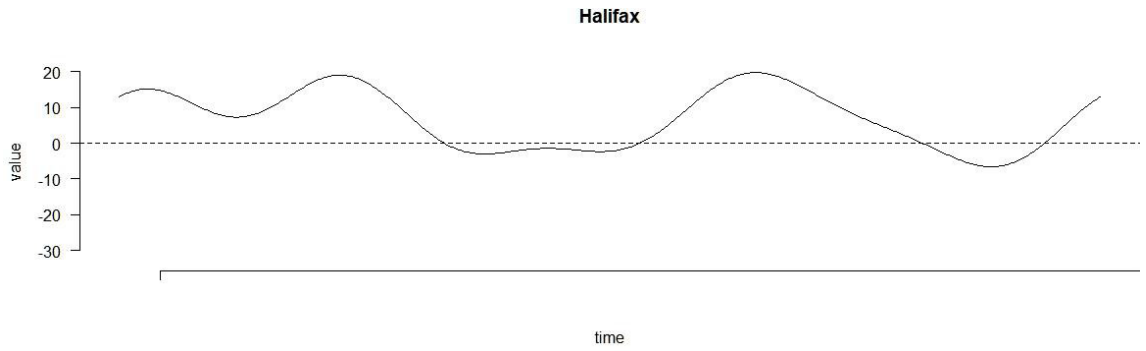

Figura 4.8: Temperaturas de la localidad de Halifax

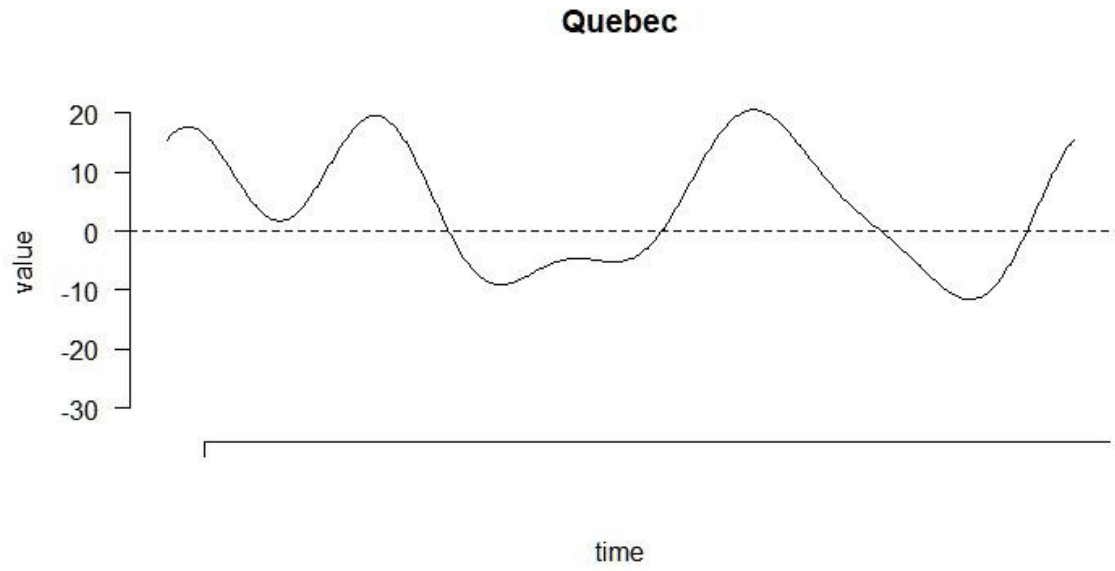

Figura 4.9: Temperaturas de la localidad de Quebec

El primer paso para trabajar con datos funcionales consiste en crear una base de funciones en el intervalo de estudio. Para ello utilizamos la función de R denominada: "create.fourier.basis(rangeval,nbasis,period)"

Donde:

rangeval: es el rango de valores que toma la variable tiempo. En este caso de 0 a 12 meses.

nbasis: corresponde al número *K* de funciones en la base *B*. En este caso K 11, no mayor al número de observaciones temporales *N*. *K* debe ser un número impar, caso contrario es aproximado al número impar superior, a fin de preservar la paridad de senos y cosenos.
period: 12 meses.

Al calcular las componentes principales funcionales por medio de la función "pca.fd" de la librería "fda". Se tiene que las tres primeras componentes principales funcionales explican el 97.8% de la variabilidad total.

	COMPF1	**COMPF2**	**COMPF3**	**TOTAL**
% VAR	87.4%	8.2%	2.2%	**97.8%**

Tabla 4.2: Varianza explicada por las 3 primeras componentes principales funcionales

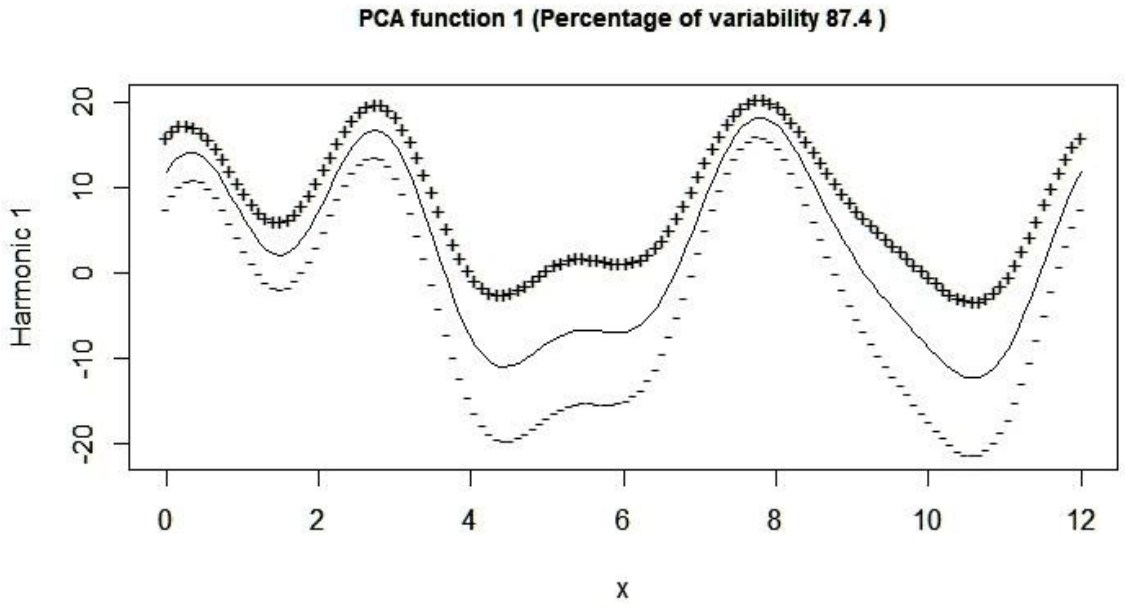

Figura 4.10: Primera componente principal funcional clásica

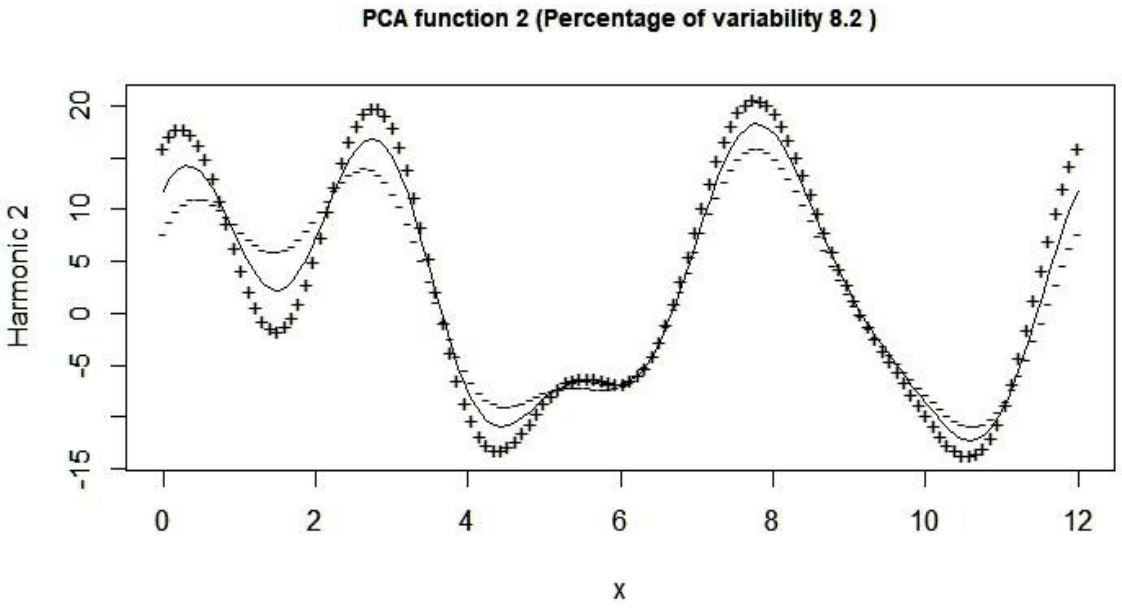

Figura 4.11: Segunda componente principal funcional clásica

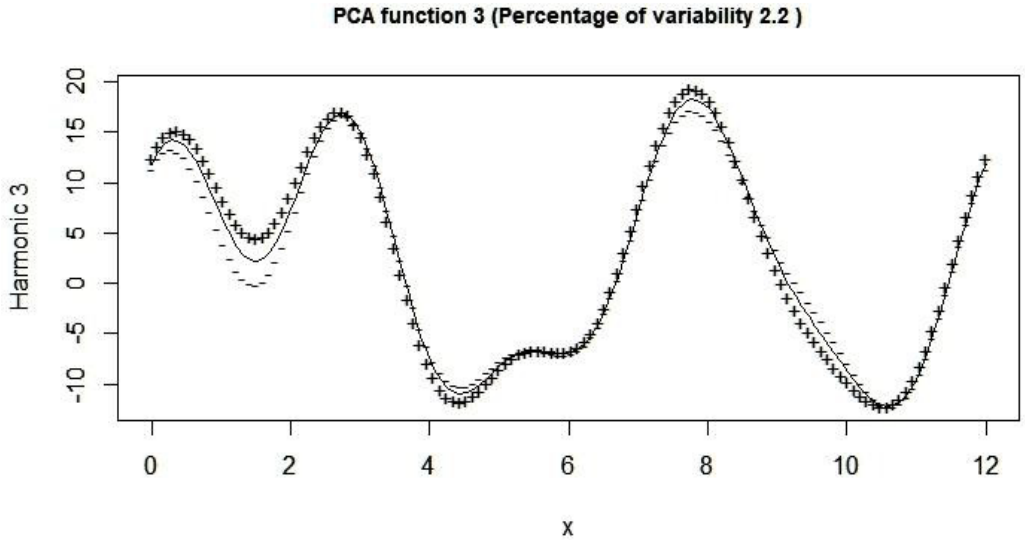

Figura 4.12: Tercera componente principal funcional clásica

Del análisis de las dos primeras componentes principales funcionales se aprecia que las temperaturas más bajas ocurren en los meses 1 y 2, es decir en enero y febrero. También en los meses de 10 y 11, que corresponden a octubre y noviembre. Mientras que las temperaturas más altas se dan en los meses de marzo y agosto.

La matriz de cargas de las primeras siete componentes principales funcionales se muestra en la Tabla 4.3:

	COMP1	**COMP2**	**COMP3**	**COMP4**	**COMP5**	**COMP6**	**COMP7**
St. Johns	0.1252	0.0921	0.2789	0.0757	0.1949	0.1605	0.0082
Halifax	0.1570	0.1250	0.1980	0.0214	0.0329	0.0650	0.0372
Sydney	0.1480	0.1078	0.3031	0.0043	0.1080	0.1340	0.0081
Yarmouth	0.1353	0.1503	0.1972	-0.0241	0.0893	-0.0920	0.2536
Charlottvl	0.1613	0.0984	0.2296	-0.0413	0.0199	0.1014	-0.0429
Fredericton	0.1736	0.0984	0.0679	-0.0338	-0.1188	0.1452	-0.2293
Scheffervll	0.1581	-0.1805	0.1224	-0.1401	0.0960	0.2900	-0.0767
Arvida	0.1845	0.0394	0.0139	-0.1982	-0.1575	0.1179	-0.0704
Bagottville	0.1823	0.0158	-0.0005	-0.1982	-0.1465	0.1933	-0.0310
Quebec	0.1819	0.0641	0.0326	-0.1060	-0.1925	0.0093	-0.1368
Sherbrooke	0.1717	0.0685	0.0174	-0.1610	-0.1853	-0.0111	-0.0154
Montreal	0.1890	0.1190	0.0315	-0.1067	-0.2160	-0.0428	0.0036
Ottawa	0.1899	0.1102	0.0088	-0.0924	-0.2192	-0.0441	-0.0063
Toronto	0.1758	0.1512	0.0949	-0.0104	-0.1708	0.0122	0.2051
London	0.1757	0.1526	0.0838	-0.0394	-0.1941	-0.0462	0.2724
Thunderbay	0.1747	0.0234	-0.0154	-0.1199	0.1477	0.1558	-0.0669
Winnipeg	0.2050	0.0163	-0.1266	-0.1427	0.0364	-0.1021	0.0582
The Pas	0.1965	-0.0533	-0.1163	-0.0903	0.1379	-0.0340	-0.1917
Churchill	0.1672	-0.2409	0.1941	-0.1075	0.5222	-0.0624	0.1166
Regina	0.1944	0.0255	-0.1550	-0.0253	0.0516	-0.0729	0.0150
Pr. Albert	0.1938	-0.0297	-0.1981	-0.1101	0.0673	-0.2259	0.0119
Uranium Cty	0.2015	-0.1452	-0.1316	-0.1578	0.0228	-0.1482	-0.2376
Edmonton	0.1681	0.0177	-0.1764	0.0079	0.0843	-0.3586	0.0460
Calgary	0.1523	0.0683	-0.0958	0.1808	0.1256	-0.1748	-0.1766
Kamloops	0.1696	0.1961	-0.1452	0.3093	0.0656	0.2764	-0.0904
Vancouver	0.1229	0.2394	0.0324	0.3216	0.0255	-0.1096	-0.0015
Victoria	0.1141	0.2322	0.0437	0.3033	0.0237	-0.1682	0.0728
Pr. George	0.1434	0.0693	-0.1693	0.1796	0.1782	0.1515	-0.2097

Pr. Rupert	0.0960	0.1662	0.0681	0.2265	0.1938	-0.1431	0.0942
Whitehorse	0.1579	-0.0650	-0.2352	0.1330	0.2122	0.2136	-0.0347
Dawson	0.2057	-0.1888	-0.4447	-0.0101	0.0119	0.2957	0.5992
Yellowknife	0.2077	-0.1979	-0.0625	0.0218	-0.0673	-0.2373	-0.3194
Iqaluit	0.1367	-0.3076	0.3218	-0.1224	0.0951	-0.2929	0.1893
Inuvik	0.1939	-0.3255	0.0232	0.4250	-0.3596	-0.1339	0.1085
Resolute	0.1226	-0.5056	0.2336	0.3465	-0.1733	0.2003	-0.0907

Tabla 4.3: Scores en el nuevo sistema referencial

El diagrama de dispersión de las localidades en el espacio generado por las dos primeras componentes principales funcionales se muestra en la figura 4.14. Aquí, el eje horizontal representa a la primera componente principal funcional y el eje vertical a la segunda componente principal funcional.

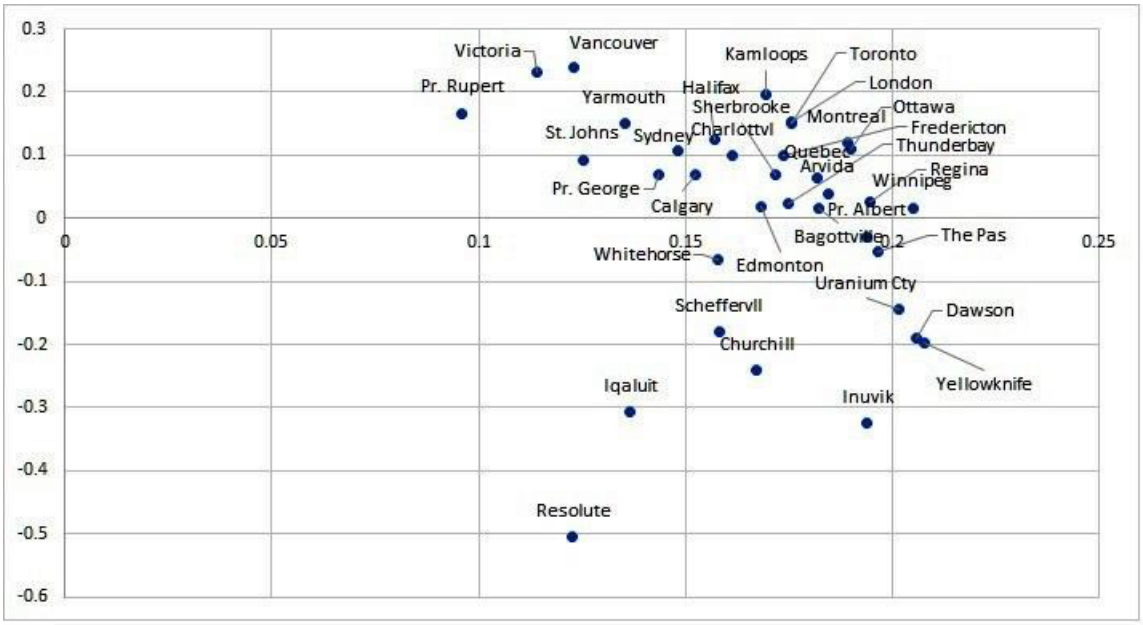

Figura 4.13: Diagrama de scores en el primer plano principal funcional

En el presente estudio nos vamos a enfocar en los scores, es decir en el espacio de los individuos (ciudades). Como se aprecia en la figura 4.13, no se puede interpretar fácilmente las posiciones de los individuos en el nuevo sistema referencial. Nos interesa encontrar una estructura de ejes disjuntos en el espacio de los individuos sin perder el orden temporal en que fueron tomados los datos. Por esta razón, se va a aplicar el algoritmo PSO

DFPC con el objetivo de obtener una mejor interpretación de las ciudades en el plano principal generado por las dos primeras componentes disjuntas. Por esta razón, para que estén todas las ciudades consideradas en el estudio en la proyección sobre el primer plano generado por los ejes disjuntos vamos a considerar Q=2, es decir, dos componentes disjuntas. Para ello aplicamos el algoritmo CBPSO DC. Necesitamos la matriz $\mathbf{\Psi}$, que contiene los coeficientes de la base de Fourier que construimos en el intervalo muestral. Para obtenerla recurrimos a la función "eval.basis". Luego obtenemos la matriz de coordenadas de las funciones de ajuste $x(t)$, en la base elegida, notada por $\mathbf{C}$:

$$\mathbf{C} \quad \left(\mathbf{\Psi}^{\dagger}\mathbf{\Psi}\right)^{-1}\mathbf{\Psi}^{\dagger}\mathbf{y}$$

Tal como se explicó en la sección 4.3.4.

Al aplicar el algoritmo CBPSO DC a la matriz de coordenadas $\mathbf{C}$ se obtiene:

	COMPF1	**COMPF2**	**TOTAL**
% VAR	58.4%	33.5%	**91.9%**

Tabla 4.4: Varianza explicada por las dos primeras componentes disjuntas

Es decir, que en conjunto las dos primeras componentes principales funcionales disjuntas explican aproximadamente el 92% de la información contenida en la muestra. Hay una pérdida del 3.7% de información al usar las componentes funcionales disjuntas. Lo que tiene sentido, puesto que las componentes disjuntas están sometidas a más restricciones que las componentes principales clásicas.

Si comparamos los porcentajes de explicación de las componentes funcionales clásicas con las disjuntas la razón es de 0.9612, lo que implica que, al usar el plano generado por las dos primeras componentes principales funcionales disjuntas, en vez de las clásicas, se pierde tan sólo el 3.8% de información.

Lo que es llamativo es la redistribución de la variabilidad entre las primeras dos componentes disjuntas.

	CPF1	CPF2	TOTAL
CLÁSICAS	87.4%	8.2%	95.6%
DISJUNTAS	58.4%	33.5%	91.9%

Tabla 4.5: Comparación de la varianza explicada

La matriz de cargas para las dos primeras componentes principales disjuntas está dada en la tabla 4.6, y su interpretación gráfica en la figura 4.14.

178	COMPF1	COMPF2
St..Johns	0.1528	0.0000
Halifax	0.1932	0.0000
Sydney	0.1803	0.0000
Yarmouth	0.1745	0.0000
Charlottvl	0.1928	0.0000
Fredericton	0.2058	0.0000
Scheffervll	0.0000	0.2790
Arvida	0.2064	0.0000
Bagottville	0.1995	0.0000
Quebec	0.2082	0.0000
Sherbrooke	0.1981	0.0000
Montreal	0.2263	0.0000
Ottawa	0.2256	0.0000
Toronto	0.2183	0.0000
London	0.2185	0.0000
Thunderbay	0.1927	0.0000
Winnipeg	0.2238	0.0000
The.Pas	0.2014	0.0000
Churchill	0.0000	0.3232
Regina	0.2142	0.0000
Pr..Albert	0.2029	0.0000
Uranium.Cty	0.0000	0.3076
Edmonton	0.1842	0.0000
Calgary	0.1769	0.0000
Kamloops	0.2198	0.0000
Vancouver	0.1780	0.0000
Victoria	0.1671	0.0000
Pr..George	0.1674	0.0000
Pr..Rupert	0.1351	0.0000
Whitehorse	0.0000	0.2138
Dawson	0.0000	0.3365
Yellowknife	0.0000	0.3444
Iqaluit	0.0000	0.3265
Inuvik	0.0000	0.4009
Resolute	0.0000	0.4221

Tabla 4.6: Scores en el nuevo sistema referencial disjunto

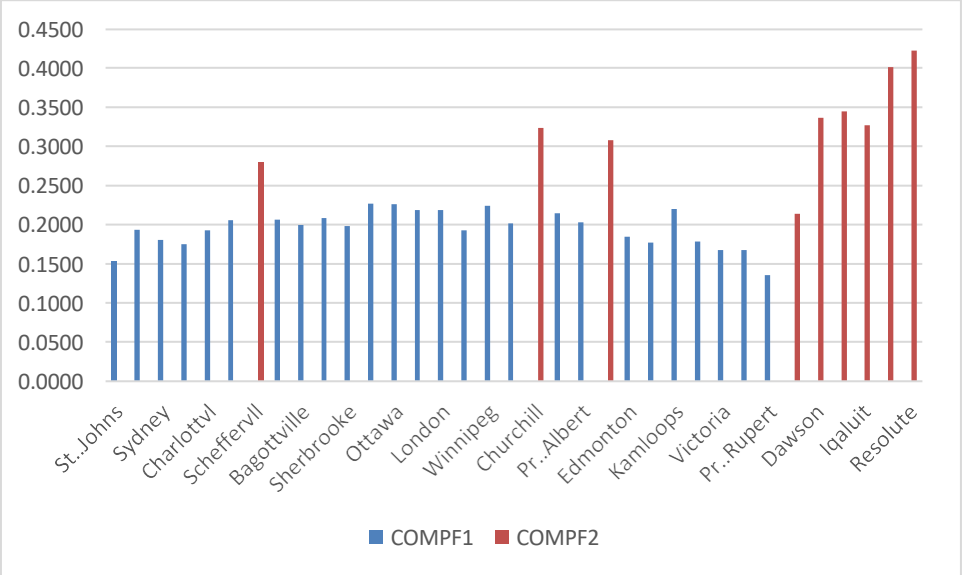

Figura 4.14: Gráfico de los scores de la tabla 4.6

Con respecto a la aplicación a datos reales, las componentes disjuntas clasifican a las localidades en dos clases: La primera componente disjunta contiene las localidades con clima más suave que se encuentran bajo la latitud 60°. Mientras que la segunda componente separa a las localidades con clima más frío. Todas las localidades que están indicadas en la segunda componente disjunta, a excepción de Schefferville, se encuentran sobre la latitud 60°. El caso de Schefferville, cuya latitud es de 54°47', se puede explicar por su altitud a nivel del mar que es de 533 metros, lo que le confiere un clima más frío.

Capítulo 5

CONCLUSIONES

Al término de este trabajo de investigación, en el que se ha implementado un nuevo método de optimización podemos concluir lo siguiente:

1. Debido a que las componentes son de norma uno y disjuntas, la matriz de cargas factoriales es ortogonal, lo que se puede interpretar como una rotación del espacio de individuos. Esto permite conservar la inercia original en el espacio rotado, y de esta forma utilizar, de manera natural, la varianza de las nuevas variables como un indicador de la varianza explicada por las componentes disjuntas.

2. Al reemplazar la búsqueda local utilizada en el algoritmo Disjoint PCA Clásico por la búsqueda inteligente del algoritmo CBPSO DC se explora de mejor forma el conjunto de soluciones factibles, lo que permite alcanzar soluciones de alta calidad sin importar el tamaño de la matriz. Por otro lado, las partículas tienen memoria compartida lo que les permite obviar rutas que conducen a soluciones que representan óptimos locales, esto último se aprecia en la calidad de las soluciones conseguidas en los ejemplos mostrados.

3. En una buena parte de la literatura consultada, cuando se trabaja con un problema de optimización con restricciones, las partículas que abandonan el conjunto de soluciones factibles son desechadas, perdiéndose toda la información histórica que tenían acumulada. Por esta razón se genera un número suficientemente grande de partículas, como previsión a las que van a abandonar el conjunto factible S.

4. En este trabajo se propone un método que obliga a que las partículas no abandonen el conjunto de soluciones factibles, sino que modifiquen su trayectoria regresando a él siempre que lo abandonen, lo que permite que en todo momento esté disponible la información histórica que han acumulado, además el número de partículas no debe ser excesivamente grande, lo que contribuye a disminuir el tiempo de procesamiento.

5. Hay que mencionar que el fit (o ajuste) obtenido con las componentes principales clásicas es ligeramente mejor que el fit obtenido con las componentes principales disjuntas. No obstante, lo que se pierde en fit se gana en calidad de interpretación.

6. Los algoritmos DPCA y CBPSO DC alcanzan la mejor solución con tasas de éxito del 100% cuando el conjunto de datos tiene pocas variables e individuos.

7. Cuando el conjunto de datos tiene muchas variables o individuos (como es el caso del ejemplo The General Self-Eficacy Scale, en el hay 19719 individuos) la tasa de éxito del algoritmo DPCA disminuye ostensiblemente a un 57%, mientras que la tasa de éxitos del algoritmo CBPSO DC se mantiene en el 100%.

8. El algoritmo CBPSO DC muestra ser bastante versátil en sus aplicaciones. Pues como se muestra en el ejemplo de datos sobre medio ambiente nos ayuda a determinar un subconjunto de variables a considerarse, disminuyendo los costos de estudios a gran escala.

9. El algoritmo CBPSO DC puede utilizarse como un medio para validar cuestionarios de forma rápida, efectiva y sobre todo de fácil interpretación. Con lo que sería una excelente contribución a los investigadores que trabajan en Ciencias Sociales, Ciencias Económicas y de la Salud.

10. Al trabajar con la optimización por enjambres dentro del contexto multivariante se presentan varias líneas de investigación futuras. En particular hay tres que vamos a mencionar: la primera es inherente a la naturaleza del método de optimización utilizado. Al ser un algoritmo de búsqueda evolutivo, no existe garantía de que la solución obtenida corresponda al óptimo global buscado, por esta razón nos referimos como la mejor solución.

11. Si bien experimentalmente las soluciones encontradas son de alta calidad, es decir tienen un buen ajuste con los datos originales, la velocidad y las rutas de las búsquedas del óptimo por el enjambre dependen de varios parámetros, como la inercia, los pesos cognitivo y social, etc., valores que no han sido investigados y que permitirán obtener un menor error de aproximación.

12. La segunda línea de investigación, en la que ya se está trabando, es sobre la utilización de la SVD en cada iteración del algoritmo.

13. La tercera línea de investigación es sobre la factibilidad de extender el algoritmo CBPSO DC a tensores o tablas de tres vías. Esta línea de investigación ya ha sido abordada con resultados prometedores.

14. Calcular la SVD de varias matrices en cada iteración consume tiempo de procesamiento (que se acrecienta al aumentar el número de individuos y variables) y constituye un cuello de botella en que ralentiza la búsqueda del óptimo, incluso utilizando PSO. Una futura línea de investigación es la de encontrar alguna alternativa para dirigir al enjambre sin utilizar la SVD. Una alternativa puede ser mediante el filtro de Kalman.

15. En el caso matricial el ambiente de los datos es un espacio de dimensión finita, el espacio $\mathbb{R}^n$. En el caso de datos funcionales es un espacio de dimensión infinita. El supuesto fundamental de la estadística de datos funcionales es que las variables aleatorias funciones se encuentran en el denominado espacio $L^2(T)$. Al tratarse de un espacio de Hilbert podemos extrapolar muchas de las propiedades geométricas de $\mathbb{R}^n$, facilitando la interpretación estadística de los datos funcionales.

16. Por otro lado, el espacio $L^2(T)$ es el espacio de la funciones cuadrado integrables, lo que corresponde a los procesos estocásticos de segundo orden. Además, se asume que son procesos continuos en media cuadrática.

17. Si bien la teoría para manejar datos funcionales es bastante exigente, al final se llega a un resultado simple. Los teoremas de Mercer y de Karhunen-Loève hacen posible mostrar que el análisis de componentes principales funcionales es equivalente al análisis de multivariante de componentes principales de una matriz.
18. Otro problema a enfrentar es la variabilidad inherente de los datos funcionales, que en muchos casos se incrementa debido a los errores de medición. Para paliar este problema se procede a ajustar los datos discretos utilizando una base de funciones suaves como series de Fourier o trazadores cúbicos. En el caso que se ha presentado en el estudio se utilizó una base de Fourier y el ajuste se realizó por mínimos cuadrados.

19. El análisis de datos funcionales presenta varios desafíos y problemas abiertos. El principal de ellos consiste en obtener las funciones propias a partir de las ecuaciones de Fredholm. Actualmente sólo existen soluciones para ciertos casos particulares. Por esta razón el FPCA discretiza los datos funcionales expresándolos como combinaciones lineales de determinada base de funciones, elegida de acuerdo a la naturaleza de los datos.

20. Si bien se asume que los datos son funciones cuadrado integrables, en realidad de lo que se dispone, en la mayoría de casos, es de mediciones discretas de estas funciones. Es decir, se tienen que interpolar o aproximar mediante funciones que pertenezcan a $L^2(T)$, aquí existe otro rubro de investigación.

ARTÍCULOS PUBLICADOS

Stochastic Environmental Research and Risk Assessment
https://doi.org/10.1007/s00477-020-01961-3

ORIGINAL PAPER

A new principal component analysis by particle swarm optimization with an environmental application for data science

John A. Ramirez-Figueroa[1,2] · Carlos Martin-Barreiro[1,2] · Ana B. Nieto-Librero[1,3] · Victor Leiva[4] · M. Purificación Galindo-Villardón[1,3]

Accepted: 11 December 2020

Abstract
In this paper, we propose a new method for disjoint principal component analysis based on an intelligent search. The method consists of a principal component analysis with constraints, allowing us to determine components that are linear combinations of disjoint subsets of the original variables. The effectiveness of the proposed method contributes to solve one of the crucial problems of multivariate analysis, that is, the interpretation of the vectorial subspaces in the reduction of the dimensionality. The method selects the variables that contribute the most to each of the principal components in a clear and direct way. Numerical results are provided to confirm the quality of the solutions attained by the proposed method. This method avoids a local optimum and obtains a high success rate when reaching the best solution, which occurs in all the cases of our simulation study. An illustration with environmental real data shows the good performance of the method and its potential applications.

Keywords Constrained binary particle swarm optimization · Data mining · Disjoint principal components · Evolutionary computation · R software · Singular value decomposition

1 Introduction

Most of multivariate techniques are based on the singular value decomposition (SVD) of the data matrix. This decomposition allows us to find a system of orthogonal principal axes (or components) that correspond to the directions of maximum variance (Beaton et al. 2014). The vectors corresponding to these new principal components are linear combinations of the original variables so that such axes are latent variables. Then, the principal component analysis (PCA) is performed by SVD, with PCA being the simplest of the eigenvectors-based multivariate techniques.

Each principal axis is related to a principal vector and the values of its coordinates give practical sense to the components in the context of the research (Hastie et al. 2009). This task can become a problem if a large number of coordinates have equal absolute values, with no loads close to zero, making it difficult to characterize the axis corresponding to the latent variable to be considered. If each of the principal axes is a linear combination of a few original variables, its interpretation is simpler within the context. Several multivariate techniques have been developed to achieve this goal. For example, the technique proposed by Vines (2000) consists of obtaining directions that are represented by vectors of integers with several of their coordinates equal to zero. The sparse PCA proposed by Zou et al. (2006) seeks factorial axes with few non-zero loads. The decomposition proposed by Mahoney and Drineas (2009) represents the data matrix as a product of three low-rank matrices. These techniques above mentioned permit us to express the principal axes based on a reduced number of columns (variables) and/or rows (individuals or objects).

✉ Victor Leiva
victorleivasanchez@gmail.com
http://www.victorleiva.cl

[1] Department of Statistics, Universidad de Salamanca, Salamanca, Spain

[2] FCNM, Universidad Politécnica ESPOL, Guayaquil, Ecuador

[3] Institute of Biomedical Research of Salamanca, Salamanca, Spain

[4] School of Industrial Engineering, Pontificia Universidad Católica de Valparaíso, Valparaíso, Chile

Published online: 02 January 2021

Another proposal is to use disjoint components that, in addition to characterizing a component as a linear combination of a few original variables, they do not appear in the other components, justifying the name disjoint. Vigneau and Qannari (2003) performed hierarchical clustering to relate a latent variable to each previously obtained cluster, proposing a disjoint PCA. A different approach to that used by Vigneau and Qannari (2003) was presented by Vichi and Saporta (2009), determining the latent variables by means of a PCA, named clustering and disjoint PCA (CDPCA). The CDPCA consists of sequentially applying cluster analysis and PCA by means of an alternating least squares algorithm. Macedo and Freitas (2015) proposed an algorithm for the CDPCA.

Another approach was proposed by Nieto-Librero et al. (2017) corresponding to the clustering and disjoint HJ biplot technique (Frutos et al. 2014). Note that a dispoint PCA facilitates the interpretation of the results, without the need of clusters to characterize them. Nieto-Librero (2015) developed a disjoint HJ biplot technique and its calculation algorithm is based on the approach proposed by Vichi and Saporta (2009) and the algorithm presented in Macedo and Freitas (2015). Ferrara et al. (2016) introduced the three following techniques to obtain disjoint components: (i) stepwise PCA; (ii) constrained PCA; and (iii) disjoint PCA, where (iii) was derived from the CDPCA. Freitas et al. (2020) compared two optimization methods for the CDPCA algorithm, one based on alternating least squares and the other one based on two-step semidefinite programming. Lou et al. (2020) proposed a method to perform PCA selecting the number of positions different from zero using particle swarm optimization (PSO).

PSO has been applied to control systems, data mining, graphical computing, neural networks, scheduling problems and robotics (Alatas and Akin 2008; Vasile and Buiu 2011). In multivariate analysis, Voss (2005) utilized PCA with PSO based on the system of principal axes being moved to the swarm, with PCA being used to determine the directions of these axes in each iteration. Chu et al. (2011) introduced a method that employs PCA to handle the bounds of the feasible region. Ma and Ji (2012) utilized PCA to reduce dimension, selecting the most important principal components and to then apply PSO. Similarly, Zhao et al. (2014) used PCA to determine efficient directions to be employed in PSO. Song et al. (2017) utilized PSO for global optimization in the presence of discontinuity. Other applications of PSO are in the cluster analysis, particularly in the detection of centroids and in the selection of the number of clusters (Van der Merwe and Engelbrecht 2003; Esmin and Matwin 2012; Gajawada and Toshniwal 2012; Wang et al. 2018). In the articles cited above, with the exception of the work on cluster analysis, the multivariate techniques utilize PSO to improve its performance in prior or posterior stages.

The present work is based on the disjoint HJ biplot technique, introducing a different approach in how the optimization is performed. Here, a new paradigm in the optimization of data science problems is exhibited to minimize the objective function using an algorithm of PSO of binary type with constraints. This technique is named here as constrained binary particle swarm optimization (CBPSO), where the term binary is due to the use of binary matrices in the PSO. Optimization by PSO is an emerging technique of evolutionary computation based on stochastic optimization and inspired by the behavior of biological species, such as starlings (Sangwook et al. 2008).

The objective of the present article is to propose a new methodology that combines PSO and PCA. Specifically, we use PSO in the solution of the optimization problem inherent to the calculation of the disjoint principal components. In addition, a new proposal is also presented here to compute the eigenvalues associated with the variability of each of the new principal axes, enabling us to construct indicators of the explicability percentages of each disjoint axis obtained.

The rest of his paper is organized as follows. Section 2 describes a dispoint PCA algorithm developed from HJ biplot, which we name DPCA in short, using an alternating least squares method. In Sect. 3, we detail the implementation of an algorithm that uses binary PSO optimization with binary constraints named here CBPSO-PCA. Section 4 summarizes the new proposed method and exposes the advantages of using PSO through simulations. In this section, we also illustrate our method with real environmental data to show its potential applications. Finally, Sect. 5 discusses the conclusions of this study and future research.

2 The DPCA method

In this section, we provide background about the DPCA. In addition, an algorithm that implements the calculation of disjoint principal components is provided.

2.1 Background

The $I \times J$ matrix $\boldsymbol{X}$ in the DPCA is expressed as in the standard PCA, that is, as the product of the transpose of the loading matrix $\boldsymbol{B}$ by the score matrix $\boldsymbol{A}$ (Jolliffe 2002, p. 31) given by

$$\boldsymbol{X} = \boldsymbol{A}\boldsymbol{B}^{\top}, \tag{1}$$

where I is the number of individuals and J is the number of variables; $\boldsymbol{A}$ is the $I \times Q$ score matrix, containing the coordinates of the individuals in the reduced Q-

dimensional space of the disjoint components; $\boldsymbol{B} = (b_{ij})$ is the $J \times Q$ component loading matrix, containing the coordinates of the variable j in the reduced Q-dimensional space and is subject to the following constraints:

(i) $\sum_{j=1}^{J} b_{jq}^2 = 1$, for $q = 1, \ldots, Q$;
(ii) $\sum_{j=1}^{J} (b_{jq} b_{jr})^2 = 0$, for $q = 1, \ldots, Q-1$ and $r = q+1, \ldots, Q$;
(iii) $\sum_{q=1}^{Q} b_{jq}^2 > 0$, for $j = 1, \ldots, J$.

Constraint (i) indicates that each column of $\boldsymbol{B}$ has norm one. Constraint (ii) expresses that two different columns of $\boldsymbol{B}$ are orthogonal, while constraint (iii) establishes that $\boldsymbol{B}$ does not have zero vectors. Since this is a low-range approximation, we have that $\boldsymbol{X}$ defined in (1) is now given by $\boldsymbol{X} = \boldsymbol{A}\boldsymbol{B}^\top + \boldsymbol{E}$, where $\boldsymbol{E}$ is the error matrix. Minimizing the Frobenius norm squared of the error matrix $\boldsymbol{E}$ is equivalent to minimizing the objective function

$$F(\boldsymbol{A}, \boldsymbol{B}) = \|\boldsymbol{X} - \boldsymbol{A}\boldsymbol{B}^\top\|^2 = \|\boldsymbol{E}\|^2, \tag{2}$$

that corresponds to the sum of residual squares. Then, we have the optimization problem stated as

$$\begin{cases} \min F(\boldsymbol{A}, \boldsymbol{B}) = \left\|\boldsymbol{X} - \boldsymbol{A}\boldsymbol{B}^\top\right\|^2 \\ \text{subject to } \boldsymbol{A}, \boldsymbol{B} \text{ as described above.} \end{cases} \tag{3}$$

Since $\boldsymbol{X}$ is the data matrix, it is a given matrix and therefore it does not vary throughout the optimization process, so that it is constant, such as its norm $\|\boldsymbol{X}\|$ is. Note that minimizing

$$\frac{\|\boldsymbol{X} - \boldsymbol{A}\boldsymbol{B}^\top\|^2}{\|\boldsymbol{X}\|^2}$$

is equivalent to minimizing $\|\boldsymbol{X} - \boldsymbol{A}\boldsymbol{B}^\top\|^2$. The function F stated in (2) is now redefined as

$$F(\boldsymbol{A}, \boldsymbol{B}) = \frac{\|\boldsymbol{X} - \boldsymbol{A}\boldsymbol{B}^\top\|^2}{\|\boldsymbol{X}\|^2}. \tag{4}$$

Thus, F redefined in (4) represents the squared relative error and is used as the objective function. Note that F measures the degree of fit of the low-range approximation. Hence, minimizing the objective function is equivalent to minimizing the relative error, and therefore a better fit corresponds to smaller values of the objective function.

In the next subsection, we explain how F is minimized to determine the number Q of principal components. This generates a Q-dimensional subspace where the original data are projected so that each original variable contributes to only one component. Note that these disjoint components are orthogonal.

2.2 Explanation of the DPCA

As mentioned, the DPCA is an adaptation of the analysis employed to construct the HJ biplot technique and the principal components (Ferrara et al. 2016). The DPCA contains the following steps:

Step 0. Let $\boldsymbol{X}$ be an $I \times J$ data matrix, with Q being chosen as the number of disjoint components to be considered. The $J \times Q$ stochastic binary matrix $\boldsymbol{V}_0$ is randomly generated by rows, such that its elements are given by

$$v_{jq} = \begin{cases} 1, \text{ if the variable } j \text{ contributes to the component } q; \\ 0, \text{ otherwise;} \end{cases}$$

satisfying the constraints

$$\sum_{q=1}^{Q} v_{jq} = 1, \quad j = 1, \ldots, J, \tag{5}$$

$$\sum_{j=1}^{J} v_{jq} > 0, \quad q = 1, \ldots, Q, \tag{6}$$

where the expression defined in (5) ensures that there is a one and nothing more than one in each row, whereas the expression defined in (6) ensures that there are no columns filled with zeros. From (5) and (6), we have that

$$\sum_{j=1}^{J} v_{jq} v_{jr} = 0, \quad q \neq r,$$

which means the columns are orthogonal.

Example 1 Let $J = 5$ and $Q = 3$. Then,

$$\boldsymbol{V}_0 = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Note that $\boldsymbol{V}_0$ is a position matrix of the original data, since it indicates how the original variables are distributed in the selected disjoint components. This must ensure that any component has all null loadings. Hence, the component with the largest number of assigned variables is randomly divided into two parts and then one of the halves is passed to the component that has the column of zeros.

Once $\boldsymbol{V}_0$ is calculated, the loading matrix $\boldsymbol{B}_0$ is obtained as follows. For each component $q = 1, \ldots, Q$, a partition matrix $\boldsymbol{W}_{0q}$ is generated from the data matrix $\boldsymbol{X}$, with $\boldsymbol{W}_{0q}$ having I rows, such as $\boldsymbol{X}$, and a number of columns corresponding to the variables indicated by the ones in the q-th column of $\boldsymbol{V}_0$, as indicated in Remark 1, where the matrix $\boldsymbol{W}$ is defined.

Remark 1 Note that the matrix $\boldsymbol{W}$ is generated from the original data $\boldsymbol{X}$ taking those columns corresponding to variables for which a one is present in the matrix $\boldsymbol{V}$ at the associated component. For instance, if for the component 1, the non-null elements of each column in the matrix $\boldsymbol{V}_0$ take the values 1, j and J, then we must take the 1-th, j-th and J-th columns. Such columns are those forming the matrix $\boldsymbol{W}_{01}$. For this matrix $\boldsymbol{W}_{01}$, we apply the HJ biplot technique calculating the coordinates for the variables. From these coordinates, we must choose the first column and such values are the coordinates of the variables 1, j and J in the first component. The remaining variables have coordinates equal to zero for this component.

Example 1 (continued) Recall that $J = 5$ and $Q = 3$. Then,

$$\boldsymbol{W}_{01} = \begin{array}{c} \begin{matrix} x_1 & x_2 \end{matrix} \\ \begin{pmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \\ \vdots & \vdots \\ x_{I1} & x_{I2} \end{pmatrix} \end{array}, \boldsymbol{W}_{02} = \begin{array}{c} \begin{matrix} x_4 \end{matrix} \\ \begin{pmatrix} x_{14} \\ x_{24} \\ \vdots \\ x_{I4} \end{pmatrix} \end{array}, \boldsymbol{W}_{03} = \begin{array}{c} \begin{matrix} x_3 & x_5 \end{matrix} \\ \begin{pmatrix} x_{13} & x_{15} \\ x_{23} & x_{25} \\ \vdots & \vdots \\ x_{I3} & x_{I5} \end{pmatrix} \end{array}.$$

Note that each $\boldsymbol{W}_{0q}$ is expressed by its SVD in the form

$$\boldsymbol{W}_{0q} = \boldsymbol{R}\boldsymbol{\Lambda}\boldsymbol{T}^{\top}, \tag{7}$$

where $\boldsymbol{R}$ is a unitary matrix, $\boldsymbol{\Lambda}$ is a rectangular diagonal matrix with non-negative real numbers on the diagonal, and $\boldsymbol{T}$ is also a unitary matrix, assuming suitable dimensions for these matrices. The diagonal entries λ_i of $\boldsymbol{\Lambda}$ are known as the singular values of $\boldsymbol{W}_{0q}$. The columns of $\boldsymbol{R}$ and $\boldsymbol{T}$ are called the left and right singular vectors of $\boldsymbol{W}$, respectively. Thus, we obtain Q SVDs of the form represented in (7).

Once the matrix $\boldsymbol{W}_{0q}$ is obtained, the matrix $\boldsymbol{B}_0$ is constructed, with the q-th column of $\boldsymbol{B}_0$ being the right singular vector corresponding to the largest singular value of the q-th SVD. This column contains the loadings corresponding to the variables considered in the q-th column of $\boldsymbol{V}_0$, with zeros in the positions that are not taken into account for the q-th column. Since Q SVDs are taken into account, the number of columns of $\boldsymbol{B}_0$ is Q. Once the matrix $\boldsymbol{B}_0$ is computed, the coordinates for the objects are obtained as $\boldsymbol{A}_0 = \boldsymbol{X}\boldsymbol{B}_0$, and then the objective function is calculated as $F_0(\boldsymbol{A}_0, \boldsymbol{B}_0)$.

Step k. It is assumed that $k-1$ steps of the algorithm have already been executed and then arrays $\boldsymbol{V}_{k-1}, \boldsymbol{A}_{k-1}, \boldsymbol{B}_{k-1}$ of F_{k-1} are available. We start by updating the matrix $\boldsymbol{V}_k$, letting the j-th row, with $j = 1, \ldots, J$, to locate a one in the q-th position of this row, for $q = 1, \ldots, Q$, and zeros in the rest of the row. Maintaining unchanged the rest of the rows of $\boldsymbol{V}_{k-1}$, we obtain a new position matrix denoted by $\widehat{\boldsymbol{V}}_{kq}$. With this position matrix, we get the partition matrix $\widehat{\boldsymbol{W}}_{kq}$, the respective $\widehat{\boldsymbol{A}}_{kq}, \widehat{\boldsymbol{B}}_{kq}$ matrices and then the objective function $\widehat{F}_{kq}(\widehat{\boldsymbol{A}}_{kq}, \widehat{\boldsymbol{B}}_{kq})$ is evaluated. The position where $\widehat{F}_{kq}(\widehat{\boldsymbol{A}}_{kq}, \widehat{\boldsymbol{B}}_{kq})$ is minimum assigns a one in that row. This process of relocating the ones in all the rows of $\boldsymbol{V}_{k-1}$ is continued and the resulting matrix is called $\boldsymbol{V}_k$. Since in each row the position containing the one runs through all the Q entries of the j-th row, the number of SVDs to determine $\boldsymbol{V}_k$ is $J \times Q$. If during the process some column of $\boldsymbol{V}_k$ is generated with all its elements equal to zero, it acts in the same way as explained in the construction of $\boldsymbol{V}_0$. Once the position matrix $\boldsymbol{V}_k$ is computed, the corresponding matrices $\boldsymbol{W}_{kq}$ are obtained and from them the loading matrix $\boldsymbol{B}_k$ is also reached in the way already indicated. Hence, the matrix $\boldsymbol{A}_k$ of scores is evaluated at $\boldsymbol{A}_k = \boldsymbol{X}\boldsymbol{B}_k$ and then $F(\boldsymbol{A}_k, \boldsymbol{B}_k) = F_k$ is calculated to check whether the stopping criterion is attained as indicated below.

Stopping criteria. Set a tolerance value $\varepsilon > 0$ and if we have $|F_k - F_{k-1}| < \varepsilon$, then the algorithm is stopped. If this is the case, the algorithm is finished and the solution is attained at this step, which is assumed as the final solution. If the stopping criterion is not reached, iterate until reaching it. Another stopping criterion is the total number of iterations to be performed. This is the stopping criterion used in the DPCA.

The matrix $\boldsymbol{V}_k$ is constructed in a non-flexible way, since when descending by rows in its construction, the positions that contain the ones of each row are fixed and do not change once they are determined. Another option is to consider all possible binary matrices for $\boldsymbol{V}_k$, but the number of possible partitions can be excessively high, even for small values of J and Q, as shown in Table 1 assuming Example 1.

Table 1 Number of operations to be performed for Example 1

J	$Q = 2$	$Q = 3$
10	$\lvert S\rvert = 1\,022$	$\lvert S\rvert = 55\,980$
15	$\lvert S\rvert = 32\,766$	$\lvert S\rvert = 14\,250\,606$
20	$\lvert S\rvert = 1\,048\,574$	$\lvert S\rvert = 3\,483\,638\,676$
30	$\lvert S\rvert = 1\,073\,741\,822$	$\lvert S\rvert = 2.058879109 \times 10^{14}$

2.3 The algorithmic form of the DPCA

Algorithm 1 summarizes the DPCA.

Algorithm 1 The DPCA procedure

1: Read $\boldsymbol{X}$, Q, `nIter` (number of iterations), and the tolerance ε.
2: Initialize with $\boldsymbol{V}_0$ and $k = 0$.
3: Iterate
  3.1: Setting $k = k + 1$.
  3.2: Updating $\boldsymbol{V}_k$.
  3.3: Obtaining the partition matrices $\boldsymbol{W}_{kq}$, for $q = 1, \ldots, Q$.
  3.4: Computing the SVD of $\boldsymbol{W}_{kq}$, for $q = 1, \ldots, Q$.
  3.5: Constructing $\boldsymbol{B}_k$ and $\boldsymbol{A}_k$.
  3.6: Calculating $F_k = F(\boldsymbol{A}_k, \boldsymbol{B}_k)$.
4: Repeat the procedure until $|F_k - F_{k-1}| < \varepsilon$.
5: Report the results for $\boldsymbol{A}_k$ and $\boldsymbol{B}_k$.

3 The CBPSO-PCA

As mentioned in Sect. 1, the PSO is an optimization technique where individuals (particles) move in the space of feasible solutions of the objective function, communicating with each other in search of the best solution (optimal approximation). The generic PSO method is presented next.

3.1 The PSO method

The PSO has the following individual and social behaviors:

(i) Particles or individuals are attracted to the optimum.
(ii) At any moment, individuals know their closeness to the optimum. The closeness is estimated through the so-called fit and is the value assigned by the objective function to the position where the particle is located.
(iii) Each particle or individual remembers its closest position to the food (the optimum). This is the individual's historical knowledge.
(iv) Each particle shares its information about its closest position to the optimum with the particles closest to it, which is the historical knowledge of its neighborhood.

Through two rules of interaction, particles adapt their behavior to that of the most successful particles in their environment (Imran et al. 2013). The PSO method allows the particles to explore the set of feasible solutions S in search of the optimum. The initial particle population is kept constant through the search process. At the instant of time k, each particle has a $\texttt{position}_k$, a $\texttt{velocity}_k$, and a value of fit F_k. The PSO method iterates at each instant of time k for each particle as follows:

(i) Local information ($\texttt{linformation}_k$): The current position where the particle reaches its best fit, that is, the smallest value of F_k for the particle.
(ii) Global information ($\texttt{ginformation}_k$): The position where the neighborhood reaches the best fit, that is, the smallest value of F_k for the swarm.

Each particle updates its position according to

$$\texttt{position}_{k+1} = \texttt{position}_k + \texttt{velocity}_{k+1}, \tag{8}$$

where

$$\begin{aligned}\texttt{velocity}_{k+1} =& \texttt{inertia}_k + \texttt{linformation}_k \\ &+ \texttt{ginformation}_k,\end{aligned} \tag{9}$$

with $\texttt{inertia}_k$ being the component responsible for keeping the particle moving at the same direction that it is moving until k.

Remark 2 Note that, in the PSO context, inertia allows a better exploration of feasible solutions to be obtained. This inertia must not be confused with the inertia in the context of multivariate analysis, in which it is a measure of the statistical variability of the data set. It must be emphasized that PSO has absolutely no guarantee to find a global optimum.

3.2 Explanation of the CBPSO-PCA

Recall that particles are represented by binary position matrices of the form $\boldsymbol{V}$ as defined in Sect. 2. In each iteration, when updating the position, the particle leaves the set of feasible solutions S, since although the position matrix is binary, the velocity matrix is not. To solve the optimization problem given in (3), the particles are represented by binary matrices $\boldsymbol{V}$ satisfying the constraints defined in (5) and (6). We must find the binary matrix $\boldsymbol{V}^{\star}$ that minimizes the objective function F and therefore this is a binary optimization problem with constraints. The search for feasible solutions leads to a problem of high computational complexity (NP-hard) due to combinatorial explosion, as mentioned at the end of Sect. 2 and exemplified in Table 1. In the present article, we use constrained binary PSO to solve the combinatorial optimization problem associated.

The first step of the PSO method is to initialize the particles, which means that they are placed at a random initial position. The number of particles P is an important input parameter at this stage. The initial position of any particle is a feasible solution of the problem, that is, an element of a set of the feasible solutions S. In each iteration, the position and velocity of the particles must be updated according to the expressions given in (8) and (9). The velocity matrices always have their components in the interval $[-1,\ 1]$, as shown in the continuation of Example 1 provided below. By updating the position at the stage

$k+1$, we have a new position that is not a binary matrix nor satisfies the conditions of the feasible space.

Example 1 (continued) Recall that $J=5$ and $Q=3$. Suppose that any particle, in the stage k, has position and velocity matrices stated by

$$\texttt{position}_k = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix},$$

$$\texttt{velocity}_{k+1} = \begin{pmatrix} -0.51 & 0.75 & 0.12 \\ -0.33 & 0.24 & 0.68 \\ -0.83 & -0.45 & 0.53 \\ 0.71 & 0.58 & 0.91 \\ 0.46 & -0.79 & 0.64 \end{pmatrix}.$$

Then, when updating the position at the stage $k+1$, according to (9), we have a new position given by

$$\begin{aligned} \texttt{position}_{k+1} &= \texttt{position}_k + \texttt{velocity}_{k+1} \\ &= \begin{pmatrix} 0.49 & 0.75 & 0.12 \\ -0.33 & 0.24 & 0.68 \\ -0.83 & 0.55 & 0.53 \\ 0.71 & 0.58 & 0.09 \\ 0.46 & -0.79 & 0.64 \end{pmatrix}. \end{aligned}$$

where $\texttt{position}_k$ is not a binary matrix nor satisfies the conditions of the feasible space. Note that the $\texttt{position}_k$ and $\texttt{velocity}_k$ matrices are chosen for illustrative purposes only, showing how the particle (in the $\texttt{position}_k$) leaves the space of feasible solutions. The calculation of both matrices is explained below.

In order to avoid the problem indicated above, we propose the following procedure:

(i) In each row, the entry with the largest absolute value is selected. The positions of these rows are occupied by ones and the rest by zeros. This procedure, as opposed to choosing random ones, allows the memory of the particle to be preserved.

(ii) In the case where a column results only with zeros, choose the column with the largest number of ones, locate the one that come from the smallest absolute value, and change it to a column full of zeros. Thus, the matrix obtained satisfies the conditions of the set of feasible solutions S.

(iii) If there are two columns full of zeros, apply the same procedure, that is, choose the two ones that come from the two smallest absolute values and then pass them to the columns of zeros.

(iv) If there is a tie in the largest absolute values of a row, choose the first of them from left to right.

The points indicated in (i)–(iv) above as solution proposed convert a matrix with real components into a binary matrix in the set of feasible solutions S, and they are executed by a matrix operator denoted by O. The fit function is given by the squared relative error, which represents, as in the DPCA, the change for one unit, such as defined in Sect. 2.1. In the PSO method, there are two types of variables: individual and collective (or global). For individual variables, each particle stores its own values, but for the global variables the values are shared by all the particles. As PSO is an evolutionary algorithm, it may happen that, in two or more successive iterations, there is no change of the objective function. For this reason, the stopping criterion is the number of iterations defined by the user (Martí et al. 2009).

In the sequel, T is a matrix operator that transforms the binary matrix $\boldsymbol{V}$ into the loading matrix $\boldsymbol{B}$ as described in the step k of Sect. 2.2. It is important to emphasize that, for calculating $\boldsymbol{B}=T(\boldsymbol{V})$, the number of SVDs performed by the operator T is the same as the number Q of desired disjoint components. Next, we describe how local and global informations work to reach the best fit.

3.2.1 Local information

The local information of a particle quantifies its attraction to the position where it reaches its best fit. For a specific particle p, the following information is stored in memory:

- $\boldsymbol{V}_p$ corresponds to the binary matrix $\boldsymbol{V}$ for the particle p and represents the current position of that particle (current binary position).
- $\boldsymbol{B}_p = T(\boldsymbol{V}_p)$ is the loading matrix $\boldsymbol{B}$ for the particle p in the current position $\boldsymbol{V}_p$.
- $F(\boldsymbol{V}_p)$ corresponds to the value of the objective function evaluated at the current position $\boldsymbol{V}_p$ of the particle p, that is, $F(\boldsymbol{V}_p) = F(\boldsymbol{A}_p, \boldsymbol{B}_p)$, where $\boldsymbol{A}_p = \boldsymbol{X}\boldsymbol{B}_p$.
- $\boldsymbol{V}_p^{\star}$ is the best binary matrix $\boldsymbol{V}$ found by the particle p locally.
- $\boldsymbol{B}_p^{\star} = T(\boldsymbol{V}_p^{\star})$ represents the best loading matrix found by the particle p locally.
- $F(\boldsymbol{V}_p^{\star})$ is the value of the objective function evaluated at the best solution found by the particle p locally.
- $\texttt{velocity}_p$ is the current velocity of the particle p used to generate a disturbance of the current position of this particle to place it at a new position. Thus, $\texttt{velocity}_p$ is a $J \times Q$ matrix, and as part of the algorithm, each entry in this matrix is in the interval $[-1, 1]$.

3.2.2 Global information

The global information quantifies the attraction of the particle towards its position that obtains the best global fit. In order to make it, the following information is collected at the level of the entire swarm:

- $\boldsymbol{V}^{\star}$ is the best binary matrix $\boldsymbol{V}$ found by the swarm of particles globally.
- $\boldsymbol{B}^{\star} = T(\boldsymbol{V}^{\star})$ is the best loading matrix found by the swarm of particles globally.
- $F(\boldsymbol{V}^{\star})$ represents the value of the objective function evaluated at the best solution found by the swarm of particles globally.

As mentioned, the PSO method consists of three main steps: initialization of the particles, iteration and update. The second step of the algorithm is the most important because it corresponds to the iterations. In each iteration, all particles must move to a new position. Each position is a feasible solution to the problem, that is, it is not allowed a particle to place itself in the position of an unfeasible solution. In this stage, we have the following parameters:

- `nIter`: number of iterations.
- `minIner`: minimum inertia value, which corresponds to the value of the inertia in the last iteration.
- `maxIner`: maximum inertia value, which is the value of the inertia in the first iteration.
- `wCognition`: cognitive weight that serves to control the cognitive component of the new velocity.
- `wSocial`: social weight, which is used to control the social component of the new velocity.

To determine the new position of a particle p, it is necessary to calculate a new velocity that depends on three components: (i) the first of them is the velocity in the previous iteration that we control with inertia; (ii) the second one is the cognitive component that we handle with the cognitive weight; and (iii) the third one is the social component that we control with the social weight. The inertia decreases linearly from iteration to iteration. The iterations are initiated with `maxIner`, whereas the iterations are finished with `minIner`. Thus, for an iteration $0 \leq k \leq \texttt{nIter}$, the inertia factor value is given by

$$\texttt{finertia} = \texttt{maxIner} - \left(\frac{\texttt{maxIner} - \texttt{minIner}}{\texttt{nIter}}\right) k. \tag{10}$$

The expression defined in (10) implies that, as it is iterated, the new velocity depends less on the previous velocity, and the cognitive and social components affect it more. The value of the new velocity for a particle p in the k-th iteration is obtained by

$$\begin{aligned}\texttt{newVel}_p = &\texttt{finertia} \times \texttt{velocity}_p \\ &+ \rho_1 \times \texttt{wCognition} \times (\boldsymbol{B}_p^{\star} - \boldsymbol{B}_p) \\ &+ \rho_2 \times \texttt{wSocial}(\boldsymbol{B}^{\star} - \boldsymbol{B}_p)\end{aligned}$$

where $\rho_1, \rho_2 \in [0, 1]$ are pseudo-random numbers so that $\texttt{newVel}_p$ has a stochastic component, $\boldsymbol{B}_p^{\star}$ is the best local loading matrix, and $\boldsymbol{B}^{\star}$ is the best global loading matrix. Note that the $J \times Q$ matrix $\texttt{newVel}_p$ generally does not comply with the constraint that the entries are in the interval $[-1, 1]$. Therefore, a transformation must be applied to satisfy this constraint. For this purpose, we define the operator L as follows.

Let $z \in \mathbb{R}$ be any entry in the matrix $\texttt{newVel}_p$. The operator L is defined by a convex linear combination of the sigmoid function, a particular case of the logistic function (Nezamabadi-Pour et al. 2008; Sangwook et al. 2008) defined as

$$L(z) = 2\text{sigmoid}(z) - 1,$$

where $\text{sigmoid}(z) = (1 + \text{e}^{-z})^{-1} \in [0, 1]$ so that $L(z) \in [-1, 1]$. The operator L is applied to all entries of $\texttt{newVel}_p$ as explained above, obtaining $\texttt{velocity}_p$. Then, a new position (in floating point format) is calculated using

$$\boldsymbol{V}_p^{(\text{temp})} = \boldsymbol{B}_p + \texttt{velocity}_p. \tag{11}$$

Again, the operator L is applied to $\boldsymbol{V}_p^{(\text{temp})}$ in (11), so that its inputs are in the interval $[-1, 1]$. Now, by applying the matrix operator O to $\boldsymbol{V}_p^{(\text{temp})}$, the new binary position of the particle p is obtained in the set of feasible solutions as $\boldsymbol{V}_p = O(\boldsymbol{V}_p^{(\text{temp})})$ and $\boldsymbol{B}_p = T(\boldsymbol{V}_p)$.

Next, we summarize the steps of initialization, iteration and update of the PSO method in an algorithmic structure:

Step 1 (initialization)

1.1 For each particle $p = 1, \ldots, P$:

- 1.1.1 Generate randomly the matrix $\boldsymbol{V}_p$.
- 1.1.2 Do $\boldsymbol{B}_p = T(\boldsymbol{V}_p)$.
- 1.1.3 Compute $F(\boldsymbol{V}_p)$.
- 1.1.4 Make $\boldsymbol{V}_p^{\star} = \boldsymbol{V}_p$.
- 1.1.5 Establish $\boldsymbol{B}_p^{\star} = \boldsymbol{B}_p$.
- 1.1.6 State $F(\boldsymbol{V}_p^{\star}) = F(\boldsymbol{V}_p)$.
- 1.1.7 Express $\texttt{velocity}_p$ randomly (initial velocity).

1.2 Among all the P particles, search for the particle with the best binary initial position, that corresponds to the particle with the best fit, denoted by $p^{\star}$. Then:

- 1.2.1 Do $\boldsymbol{V}^{\star} = \boldsymbol{V}_{p^{\star}}$.
- 1.2.2 Make $\boldsymbol{B}^{\star} = \boldsymbol{B}_{p^{\star}}$
- 1.2.3 State $F(\boldsymbol{V}^{\star}) = F(\boldsymbol{V}_{p^{\star}})$.

Step 2 (iteration)

2.1 Determine the new position of a particle p.
2.2 Establish the value of the new velocity for a particle p in the k-th iteration.
2.3 Transform the entries to the interval $[-1, 1]$ using the operator L.
2.4 Calculate a new position using the expression given in (11) for $\boldsymbol{V}_p^{(\text{temp})}$.
2.5 Apply again the operator L to $\boldsymbol{V}_p^{(\text{temp})}$ so that its inputs are in the interval $[-1, 1]$.
2.5 Obtain the new binary position of the particle p in the set of feasible solutions by using the operators O and T in $\boldsymbol{V}_p^{(\text{temp})}$ to reach $\boldsymbol{V}_p = O(\boldsymbol{V}_p^{(\text{temp})})$ and $\boldsymbol{B}_p = T(\boldsymbol{V}_p)$.

Step 3 (updating)

3.1 For each iteration $k = 1, \ldots,$ `nIter` :

3.1.1 Compute `finertia`.
3.1.2 For each particle $p = 1, \ldots, P$:

3.1.2.1 Calculate $\texttt{newVel}_p$.
3.1.2.2 Do $\texttt{velocity}_p = L(\texttt{newVel}_p)$.
3.1.2.3 Make $\boldsymbol{V}_p^{(\text{temp})} = \boldsymbol{B}_p + \texttt{velocity}_p$.
3.1.2.4 State $\boldsymbol{V}_p = O(\boldsymbol{V}_p^{(\text{temp})})$.
3.1.2.5 Establish $\boldsymbol{B}_p = T(\boldsymbol{V}_p)$.
3.1.2.6 Determine $F(\boldsymbol{V}_p)$.
3.1.2.7 If $F(\boldsymbol{V}_p) < F(\boldsymbol{V}_p^{\star})$, then
A. $\boldsymbol{V}_p^{\star} = \boldsymbol{V}_p$.
B. $\boldsymbol{B}_p^{\star} = \boldsymbol{B}_p$.
C. $F(\boldsymbol{V}_p^{\star}) = F(\boldsymbol{V}_p)$.
If $F(\boldsymbol{V}_p) < F(\boldsymbol{V}^{\star})$, then
A. $\boldsymbol{V}^{\star} = \boldsymbol{V}_p$.
B. $\boldsymbol{B}^{\star} = \boldsymbol{B}_p$.
C. $F(\boldsymbol{V}^{\star}) = F(\boldsymbol{V}_p)$.

3.2 Report the best solution found.

3.3 The CBPSO-PCA algorithm

The way how the CBPSO-PCA works is summarized in Algorithm 2. The function `Step` that we see in Algorithm 2 is used to represent, in a single instruction, the movement of a particle to a new position. Every time a particle moves to the next position, the best position found by that particle and the best position found by all particles are updated, if necessary. Note that the `finertia` is updated when going from an iteration to the next. All the particles of the same iteration share the same `finertia` value. In Algorithm 2, the presence of the matrices $\boldsymbol{V}_p$, $\boldsymbol{V}_p^{\star}$, and $\boldsymbol{V}^{\star}$ is omitted to simplify the operation of the CBPSO-PCA algorithm. Here, $F(\boldsymbol{B}_p) = F(\boldsymbol{A}_p, \boldsymbol{B}_p)$.

Algorithm 2 CBPSO-PCA

1: Initiate P particles at random.
2: Do `finertia` $=$ `maxIner`.
3: For each iteration $k = 1, \ldots,$ `nIter`:
For each particle $p = 1, \ldots, P$:
Do $\boldsymbol{B}_p =$ `Step`$(\boldsymbol{B}_p, \boldsymbol{B}_p^{\star}, \boldsymbol{B}^{\star},$ `wCognition, wSocial, finertia`$)$.
If $F(\boldsymbol{B}_p) < F(\boldsymbol{B}_p^{\star})$, then do $\boldsymbol{B}_p^{\star} = \boldsymbol{B}_p$.
If $F(\boldsymbol{B}_p) < F(\boldsymbol{B}^{\star})$, then do $\boldsymbol{B}^{\star} = \boldsymbol{B}_p$.
End for each p.
Update `finertia`.
End for each k.
4: Report the obtained results.

3.4 Explained variance

Note that the CBPSO-PCA does not calculate the matrix $\boldsymbol{\Lambda}$ defined in (7). Then, the singular values of $\boldsymbol{X}$ are unknown and therefore the eigenvalues of $\boldsymbol{X}^{\top}\boldsymbol{X}$ as well. Here, we estimate the eigenvalues corresponding to each of the disjoint components by means of the calculation of the variance that the individuals exhibit in the new system of principal axes. This enables us to find the percentage of explained variance for each of the disjoint axes and the respective principal planes. Specifically, let $\boldsymbol{X}$ be a centered $I \times J$ data matrix, and let $\text{Var}(\boldsymbol{X}) = \boldsymbol{S}$ be its variance-covariance matrix. Then, the total variation of $\boldsymbol{X}$ is defined by

$$\text{trace}(\boldsymbol{S}) = \sigma_1^2 + \cdots + \sigma_J^2 = \sum_{j=1}^{J} \sigma_j^2, \tag{12}$$

where $\sigma_1^2, \ldots, \sigma_J^2$ are the variances of the J variables contained in the columns of $\boldsymbol{X}$, that is, $\text{Var}(X_j) = \sigma_j^2$. Alternatively, we have that

$$\text{trace}(\boldsymbol{S}) = \alpha_1 + \cdots + \alpha_J, \tag{13}$$

where $\alpha_1, \ldots, \alpha_J$ are the eigenvalues of $\boldsymbol{S}$.

Let $\boldsymbol{B}$ be a $J \times J$ rotation matrix, that is, $\boldsymbol{B}$ is orthogonal in the sense that $\boldsymbol{B}^{\top}\boldsymbol{B} = \boldsymbol{I}_J$, where $\boldsymbol{I}_J$ is $J \times J$ identity matrix. The column vectors $\boldsymbol{B}$ define a new rotated system of axes. Each new axis is a linear combination of the original variables representing a new variable. Let $\boldsymbol{A}$ be a rotation transformation of $\boldsymbol{X}$ given by $\boldsymbol{A} = \boldsymbol{X}\boldsymbol{B}$. Its variance-covariance matrix is stated as

$$\boldsymbol{\Sigma} = \text{Var}(\boldsymbol{A}) = \boldsymbol{B}^{\top}\boldsymbol{S}\boldsymbol{B}.$$

Since the columns of $\boldsymbol{A}$ contain the coordinates of the individuals with respect to the new system of axes, the variance of each column represents the variance of the new variables obtained by rotation $\boldsymbol{B}$. The total variation of $\boldsymbol{A}$ is defined as

$$\begin{aligned}\text{trace}(\boldsymbol{\Sigma}) &= \text{trace}\left(\boldsymbol{B}^{\top}\boldsymbol{S}\boldsymbol{B}\right)\\ &= \text{trace}\left(\boldsymbol{S}\boldsymbol{B}\boldsymbol{B}^{\top}\right)\\ &= \text{trace}\left(\boldsymbol{S}\boldsymbol{I}_J\right)\\ &= \text{trace}(\boldsymbol{S}),\end{aligned} \tag{14}$$

that is, the total variation of $\boldsymbol{X}$ and $\boldsymbol{A}$ is the same. Note that the rotation $\boldsymbol{B}$ does not affect the total variation. If $\beta_1, \ldots, \beta_J$ are the eigenvalues of the matrix $\boldsymbol{\Sigma}$ in descending order, then the variance of the j-th new variable, denoted by A_j, is $\text{Var}(A_j) = \beta_j$, for $j = 1, \ldots, J$, and

$$\text{trace}(\boldsymbol{\Sigma}) = \beta_1 + \cdots + \beta_J. \tag{15}$$

From (12), (13), (14) and (15), we have

$$\sigma_1^2 + \cdots + \sigma_J^2 = \alpha_1 + \cdots + \alpha_J = \beta_1 + \cdots + \beta_J.$$

Therefore, the coefficients β_i can be used to determine the percentages of explanation of the disjoint axes. Thus, the percentage of the total variation explained by the q-th disjoint axis is given by

$$\frac{\beta_q}{\sum_{i=1}^{J}\beta_i} \times 100\% = \frac{\beta_q}{\sum_{j=1}^{J}\sigma_j^2} \times 100\%. \tag{16}$$

The amount $\sum_{j=1}^{J}\sigma_j^2$ can be obtained directly from the matrix $\boldsymbol{X}$ and the coefficients β_j may be estimated from $\text{Var}(A_j)$. In the case $Q = J$, when applying the CBPSO-PCA algorithm to obtain the disjoint components, the loading matrix $\boldsymbol{B} = \boldsymbol{I}_J$ is obtained and then

$$\boldsymbol{A} = \boldsymbol{X}\boldsymbol{B} = \boldsymbol{X}.$$

If $Q < J$, the columns of $\boldsymbol{B}$ are disjoint with norm equal to one. Thus, they form an orthonormal set of Q vectors in $\mathbb{R}^J$, and additionally $\boldsymbol{B}^{\top}\boldsymbol{B} = \boldsymbol{I}_Q$. Hence, $\boldsymbol{B}$ can be considered as a projection matrix in the new rotated disjoint system of axes and $\boldsymbol{A} = \boldsymbol{X}\boldsymbol{B}$ is the matrix containing the coordinates of the I objects found in the vector subspace generated by the first Q disjoint components.

Note that our proposal is to use the variance of scores, that is, we can apply the indicators defined in (16). The variance of the columns of $\boldsymbol{A}$ is employed to calculate the variance explained by each disjoint axis obtained, and for each principal plane required. This criterion for calculating the variance of the columns of the scores matrix is utilized to determine the proportion of variability explained by each factor axis, both in the DPCA and CBPSO-PCA in the numerical examples of the following sections. Observe that is not required to calculate the matrix of singular values $\boldsymbol{\Lambda}$.

3.5 Application of the CBPSO-PCA algorithm

Algorithm 3 presented below is a guide to how using Algorithm 2 (CBPSO-PCA, given in Sect. 3.3) when computing the disjoint components in a practical context, that is, when a centered data matrix $\boldsymbol{X}$ is available.

Algorithm 3 Application of the CBPSO-PCA algorithm

1: Read the $I \times J$ centered data matrix $\boldsymbol{X}$.
2: Perform a usual exploratory PCA.
3: Compute the variance explained by the principal components.
4: Find the number of disjoint components Q using the investigator's criterion based on Step 3.
5: Calculate Q according to the CBPSO-PCA:
  5.1 Establishing the stopping conditions.
  5.2 Stating the number of particles, maximum, minimum inertia, and social, cognitive weights.
  5.3 Applying Algorithm 2.
6: Repeat Step 5 until finding the best fit.
7: Report the obtained results.

4 Empirical illustrations

In this section, we show the advantages of the CBPSO-PCA algorithm, illustrating with empirical data the quality of the solutions that can be found. First, we consider two simulated examples and then a real environmental data example.

4.1 Computational and simulation aspects

All computational experiments are performed on a 64-bit Windows 10 computer, 8 GB of RAM, and an Intel (R) Core (TM) i7-4510U 2-2.60 GHz processor. The algorithm is implemented employing `C#.NET` and `R`. The use of the `R` statistical software (R Core Team 2018) is important primarily for performing the SVD. More specifically, the `irlba` package is used as it provides a fast and efficient way to calculate a partial SVD for large matrices, instead of using the `svd` function of the `svd` package of `R`, that provides a generic SVD of a matrix. Communication between `C#.NET` and `R` is possible using the `R.NET` middleware, which is installed in the corresponding code project as a `NuGet` package. The data matrix is located in an `Excel` sheet and is read from `C#.NET` code at runtime using `COM+`.

The stopping criterion used in the CBPSO-PCA algorithm is the number of iterations. However, each time the algorithm finds a better solution, it is stored along with its processing time. The best solution found by the algorithm usually has a processing time less than the time needed for all iterations to run.

4.2 Generator with disjoint component structure

A simulation algorithm is constructed to randomly generate a data matrix, with an ad-hoc structure of easy interpretation. Then, when applying a dimension reduction by

disjoint principal components, the CBPSO-PCA algorithm is able to detect it.

Let $x_1, \ldots, x_p$ be the values of original p variables and $y_1, \ldots, y_q$ be the values of q latent variables ($q < p$) with disjoint structure. Consider the linear combination given by $y_i = c_{1,i}x_1 + \cdots + c_{p,i}x_p$. If the m original consecutive variables $x_j, x_{j+1}, \ldots, x_{j+(m-1)}$ are represented in the latent variable y_i, then the scalars $c_{j,i}, c_{j+1,i}, \ldots, c_{j+(m-1),i}$ are defined as independent and uniform discrete distributed random variables with support on the integers from 70 to 100. Note that the remaining scalars are defined similarly but with support on the integers from 1 to 30. This procedure is performed for each i from 1 to q. Keep in mind that each original variable must have a strong presence in one single latent variable.

Example 2 Suppose that $p = 8$, $q = 3$ and $\mathrm{U_d}(a, b)$ denotes the discrete uniform distribution, with support on the integers from a to b ($a, b \in \mathbb{N}$, $a < b$). Note that the first linear combination is stated as $y_1 = c_{1,1}x_1 + c_{2,1}x_2 + c_{3,1}x_3 + c_{4,1}x_4 + c_{5,1}x_5 + c_{6,1}x_6 + c_{7,1}x_7 + c_{8,1}x_8$. If x_1, x_2, x_3, x_4 are represented in y_1, then $c_{1,1}, c_{2,1}, c_{3,1}, c_{4,1} \overset{\text{IID}}{\sim} \mathrm{U_d}(70, 100)$ and $c_{5,1}, c_{6,1}, c_{7,1}, c_{8,1} \overset{\text{IID}}{\sim} \mathrm{U_d}(1, 30)$, where "IID' denotes "independent and identically distributed". The second linear combination is defined as $y_2 = c_{1,2}x_1 + c_{2,2}x_2 + c_{3,2}x_3 + c_{4,2}x_4 + c_{5,2}x_5 + c_{6,2}x_6 + c_{7,2}x_7 + c_{8,2}x_8$. If x_5, x_6, x_7 are represented in y_2, then $c_{5,2}, c_{6,2}, c_{7,2} \overset{\text{IID}}{\sim} \mathrm{U_d}(70, 100)$ and $c_{1,2}, c_{2,2}, c_{3,2}, c_{4,2}, c_{8,2} \overset{\text{IID}}{\sim} \mathrm{U_d}(1, 30)$. And the third linear combination is expressed as $y_3 = c_{1,3}x_1 + c_{2,3}x_2 + c_{3,3}x_3 + c_{4,3}x_4 + c_{5,3}x_5 + c_{6,3}x_6 + c_{7,3}x_7 + c_{8,3}x_8$. We only have left the original variable x_8, which must be represented in y_3. Then, we get $c_{8,3} \sim \mathrm{U_d}(70, 100)$ and

$$c_{1,3}, c_{2,3}, c_{3,3}, c_{4,3}, c_{5,3}, c_{6,3}, c_{7,3} \overset{\text{IID}}{\sim} \mathrm{U_d}(1, 30).$$

In Example 2, the first four original variables have a strong presence in the first latent variable, the next three original variables do so in the second latent variable, and the last original variable has a strong representation in the third and last latent variable. We indicate this, in a general way, by the finite succession $\{r_k\}_{k=1}^{q}$, whose elements for the previous particular case are $r_1 = 4$, $r_2 = 3$ and $r_3 = 1$. The algorithm designed simulates a matrix with n individuals and p variables. The orthonormalization Gram-Schmidt process is applied to this matrix to obtain the simulated data matrix $\boldsymbol{X}$. The algorithm that builds the $n \times p$ random matrix $\boldsymbol{X}$ should, in general, implement the mapping ϕ defined as

$$\begin{aligned} &\phi\colon \mathbb{N} \times \mathbb{N} \times \mathbb{N} \times \mathbb{N}^q \to \boldsymbol{M}_{n \times p} \\ &\left(n,\ p,\ q, \{r_k\}_{k=1}^{q}\right) \mapsto \phi\left(n,\ p,\ q, \{r_k\}_{k=1}^{q}\right) \end{aligned}$$

subject to the constraints $n \geq p$ and $q < p$. In addition, it should be satisfied that

$$p = \sum_{k=1}^{q} r_k.$$

The mapping ϕ returns an $n \times p$ matrix $\boldsymbol{X}$ that has the ad-hoc structure mentioned above, which should be detected by the CBPSO-PCA algorithm.

4.3 Simulation studies

For the first simulation, the mapping $\phi(100, 8, 3, \{4, 3, 1\})$ is executed, that is, there are 100 individuals and 8 original variables and then a 100×8 data matrix $\boldsymbol{X}$ is obtained. In addition, the latent structure is made up of three disjoint components: the first one has non-zero charges in the first four positions. The second disjoint component has non-zero charges at 5-th, 6-th and 7-th positions. The last disjoint component has a non-zero charge at the last position. The CBPSO-PCA algorithm provides the loading matrix $\boldsymbol{B}$ reported in Table 2.

The CBPSO-PCA is executed 100 times with 50 particles and 10 iterations as the stopping criterion. It should be stressed that no background process is run on the computer while the calculations are being made. In all 100 executions, the same solution is reached as shown in Table 2, needing two or three iterations at most. The CBPSO-PCA algorithm is able to detect the ad-hoc structure contained in the data. The fit obtained when applying disjoint principal components is 0.021859878 and the total variance explained by the model is 97.81%. The DPCA algorithm is also executed 100 times and the same solution given in Table 2 is reached 100% of the times. A usual PCA is also

Table 2 Loading matrix $\boldsymbol{B}$ obtained by CBPSO-PCA and DPCA algorithms with simulated data

	Y_1	Y_2	Y_3
X_1	0.43413655	0	0
X_2	0.53903007	0	0
X_3	0.56700392	0	0
X_4	0.44663027	0	0
X_5	0	0.57919604	0
X_6	0	0.56750625	0
X_7	0	0.58520817	0
X_8	0	0	1
% of EV	35.54%	33.01%	29.26%

where EV denotes the explained variance

performed on the same data matrix and the loading matrix defined in Table 3 is obtained. Note that the usual PCA does not clearly detect the underlying structure in the data. For example, what happens to the second original variable X_2 is noteworthy: it cannot be concluded with certainty in which component it has a strong presence. The same can be said of the last original variable X_8.

A second simulation study is conducted to compare the performance of the CBPSO-PCA and DPCA algorithms when the matrix size is large. The simulation algorithm implements the mapping $\phi(200, 200, 3, \{50, 70, 80\})$ and a 200×200 data matrix $\boldsymbol{X}$ with three disjoint components. After executing both algorithms 100 times each, the results obtained are summarized in Table 4. Both algorithms found the best solution, with a fit of 0.02543703 and a total explained variance of 97.46%, with different success rates. The CBPSO-PCA algorithm is executed with the same input parameters as in the first simulation study, and the best time reached is 4.2 minutes in 6 iterations. In addition, for the DPCA algorithm, the best time reached is 9.8 minutes in two iterations. The last row of Table 4 reports that the DPCA is trapped in a local optimum in 7 of the 100 executions. It is important to highlight that the DPCA algorithm does not perform an exhaustive search, since starting from the binary matrix $\boldsymbol{V}$ that it initially generates randomly, it begins to move by column the number one in all possible positions.

Note that, as the number of original variables and latent variables increases, we detect a larger processing time in this algorithm. The CBPSO-PCA algorithm is better suited for large arrays, since it has the flexibility to fit its parameters allowing a better search strategy. The DPCA algorithm has only one parameter (tolerance) that is used as a stopping criterion. With respect to the percentages of success, the DPCA algorithm has a larger probability of being trapped in a local optimum, since it begins its processing with a single binary matrix $\boldsymbol{V}$, that is modified by displacements of the one row and column, not necessarily leading to an optimal matrix. As mentioned in Macedo and Freitas (2015), the algorithm can be trapped in different local optimum so that it is recommended to execute it several times. In addition, the CBPSO-PCA algorithm starts with a matrix $\boldsymbol{V}$ for each particle, and it is the joint and intelligent search of the particles, which permits the CBPSO-PCA to have a small probability of being trapped in a local optimum.

Table 3 Loading matrix $\boldsymbol{B}$ obtained by usual PCA algorithm with simulated data

	Y_1	Y_2	Y_3
X_1	0.39512468	−0.09020236	0.14824807
X_2	0.39775809	−0.00322755	0.40788776
X_3	0.44254179	−0.17190775	0.31755350
X_4	0.35239424	−0.11448361	0.25061667
X_5	0.26682407	0.46011771	−0.25159988
X_6	0.30662152	0.39441186	−0.26207760
X_7	0.35705213	0.28282245	−0.38998176
X_8	−0.27007773	0.70847497	0.60326462
% of EV	39.81%	32.27%	27.92%

where EV denotes the explained variance

Table 4 Summary of the second simulation study

Characteristic	CBPSO-PCA	DPCA
Average execution time (in minutes)	4.6	10.2
Best execution time (in minutes)	4.2	9.8
Worst execution time (in minutes)	4.8	10.4
Success rate	100%	93%

We compare the convergence of the proposed algorithm with the usual PCA. If the mathematical optimization model for obtaining disjoint components is relaxed, eliminating precisely the constraint for the calculation of these components, the minimization model for calculating the usual principal components is obtained. Therefore, the fit of this PCA is always better than the fit of a DPCA. Figure 1 shows a black curve that relates the number of iterations of the CBPSO-PCA algorithm with the fit (value of the objective function). The gray line, parallel to the horizontal axis, shows the fit of a standard PCA. For the second simulation study, the fit value of a PCA is so small that the R software represents that fit with a zero. Observe that the best fit obtained by the CBPSO-PCA algorithm in one of the executions is 0.02543703 in 6 iterations of a maximum of 7 iterations.

4.4 Application to real environmental data

A problem that arises when conducting studies with environmental or ecological data is that of the presence of

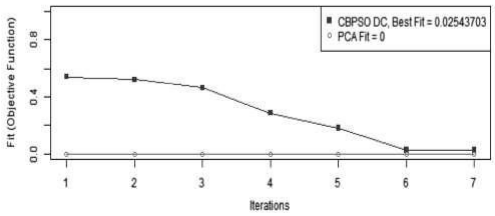

Fig. 1 Convergence analysis for the second simulation study according to the method indicated

redundant variables, that is, variables that are an exact or approximate linear combination of one or more of the remaining variables. These variables can be eliminated without losing information or with minimal loss of it, resulting in a smaller data set, reducing the costs for the study.

Eliminating redundant variables, and retaining those that provide the most information is relevant, particularly if the data set is massive, as it is often occurs in environmental studies. The PCA allows us to reduce the dimensionality of the problem under study. Since the principal components are not correlated, they remove redundant information. The problem that arises, as already indicated, is its interpretation, because the loadings do not always approach zero in absolute value. The CBPSO-PCA permits us to alleviate this problem by presenting the variables that contribute little information (to a certain principal component) with values equal to zero. Thus, we can select the variables that provide the most information to the first disjoint principal component (that is, the one that explains a high variability) and discard the variables that are found in the remaining disjoint principal components. Obviously, this decision depends on many factors such as the type of research to be carried out, the importance of the variables to be excluded and the knowledge of the researcher who collected the data.

There is an extensive bibliography in which different methodologies are presented for reducing the number of variables in a multivariate study, among which we can highlight: the seminal work of Jolliffe (1973), who used PCA through multiple linear regression, and the work of King and Jackson (1999), which is more specific, since there different methods are proposed to discard redundant variables in environmental studies. King and Jackson (1999) showed that the Broken-Stick method is the best criterion to determine the number of principal components to retain, and then select the variables to be discarded using the different models developed by Jolliffe (1973).

When the size of a data set is decreased, a stability problem arises, which consists of the concordance between the distances of the individuals in the space generated by the original variables and the space generated by the set containing the selected variables (Sorzano et al. 2014). In order to reduce the number of the original variables using PCA in ecological data to be stable, the interested reader is referred to Grossman et al. (1991), where the ratio between original variables and selected variables is considered to be less or equal than three.

In this application, we use the Broken-Stick method and the CBPSO-PCA algorithm to select the variables to be retained in an environmental study. The data set used is taken from a database corresponding to the FOREGSEuroGeoSurveys Geochemical Baseline available at http://www.gtk.fi/publ/foregsatlas.

The data correspond to samples of contents of minerals taken in the subsoil of different countries of the Union European. The list of variables is shown in Table 5. These data can be used to measure subsurface contamination and obtained from the file `C_XRF_data_2v6_8Fe-b06.xls`, available at http://weppi.gtk.fi/publ/foregsatlas/ForegsData.php.

For the analysis, the columns corresponding to the coordinates of their geographical location and the country to which they correspond have been eliminated. The data matrix consists of 19 columns or variables (the elements and chemical compounds) and 788 rows (locations where the samples are collected). The variables represent the content of mineral found in the soil at each sampled location. Since the units of measurement differ, depending on the variable, we standardize the variables for the analysis.

We proceed as follows. First, we apply a usual PCA to the data set. Second, we select Q, the number of principal components to consider using the Broken-Stick method. Third, we calculate the Q disjoint principal components with the CBPSO-PCA algorithm. And fourth, we take into account the variables that make up the first disjoint principal component. The results of the PCA of the data matrix are reported in Table 6.

Table 5 List of variables under study

Variable	Chemical compound/element	Measurement units
SIO2_SI	Silicon oxide	kg
TIO2_SI	Titanium oxide	kg
AL2O3_SI	Aluminum oxide	kg
FE2O3_SI	Iron oxide	kg
MNO_SI	Manganese oxide	kg
MGO_SI	Magnesium oxide	kg
CAO_SI	Calcium oxide	kg
NA2O_SI	Sodium oxide	kg
K2O_SI	Potassium oxide	kg
P2O5_SI	Phosphorus oxide	kg
BA_SI	Barium	mg
CR_SI	Chrome	mg
RB_SI	Rubidium	mg
SN_SI	Tin	mg
SR_SI	Strontium	mg
W_SI	Tungsten	mg
Y_SI	Yttrium	mg
ZN_SI	Zinc	mg
ZR_SI	Zirconium	mg

Figure 2 shows a graph corresponding to the Broken-tick method (black line) superimposed on a gray curve that indicates the explained variance percentage by considering the principal components indicated on the x-axis. The point cutting is close to the third principal component. For this reason, we take into account three principal components to retain (that is, $Q = 3$). These first three principal components explain 51.66 % of the total variability. Therefore, we use $Q = 3$ to apply the CBPSO-PCA algorithm, whose results are reported in Tables 7 and 8. The three disjoint components explain 46.78 % of the total variability. If we consider the first three principal components suggested by the Broken-Stick method, which explain 51.96 % of the variability, we have that the disjoint components explain 90.01 % of the variance by using the usual first three principal components. This low percentage explained by the disjoint components is due to the fact that they must satisfy a set of additional restrictions to the usual principal components; see Sect. 2. In Fig. 3, the results of Table 7 are represented, where we clearly appreciate how the original variables are distributed in the disjoint principal components, their weight and their sign.

The convergence analysis shows that the CBPSO-PCA algorithm takes six iterations to converge to the value 0.5322, whereas he CBPSO algorithm converges in just 5

Table 6 Results of the PCA of the environmental data matrix

#PC	Eigenvalue	% of EV	% of AEV
1	5.095	26.815	26.815
2	2.941	15.477	42.292
3	1.838	9.674	51.966
4	1.488	7.833	59.799
5	1.301	6.845	66.644
6	1.025	5.397	72.041
7	0.904	4.759	76.800
8	0.797	4.193	80.993
9	0.722	3.801	84.794
10	0.650	3.423	88.217
11	0.489	2.572	90.789
12	0.412	2.166	92.955
13	0.387	2.039	94.994
14	0.325	1.708	96.703
15	0.267	1.408	98.110
16	0.125	0.657	98.768
17	0.105	0.554	99.322
18	0.079	0.417	99.739
19	0.050	0.262	100.000

where *PC* principal component, *EV* explained variance, and *AEV* accumulated explained variance

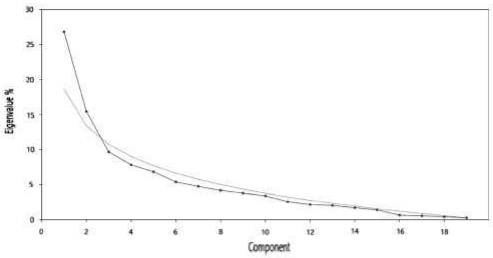

Fig. 2 Broken-Stick plot with environmental data (in black) and % of explained variance by the indicated principal component (in gray)

Table 7 Results of the CBPSO-PCA for the indicated variable and component with environmental data

	# disjoint principal component		
Variable	1	2	3
SIO2_SI	0	0	0.57973524
TIO2_SI	0.44020387	0	0
AL2O3_SI	0	−0.46399195	0
FE2O3_SI	0.49562162	0	0
MNO_SI	0.44137161	0	0
MGO_SI	0	0	−0.34305531
CAO_SI	0	0	−0.55125267
NA2O_SI	0	−0.28220297	0
K2O_SI	0	−0.49633089	0
P2O5_SI	0.27387266	0	0
BA_SI	0	−0.43570013	0
CR_SI	0.18954184	0	0
RB_SI	0	−0.47588306	0
SN_SI	0	−0.15767984	0
SR_SI	0	0	−0.34117431
W_SI	0	−0.13253821	0
Y_SI	0.42150555	0	0
ZN_SI	0.27779775	0	0
ZR_SI	0	0	0.35488125
% of EV	17.34%	16.86%	12.58%

where EV denotes the explained variance

Table 8 Results of the CBPSO-PCA of the environmental data matrix

#PC	% of EV	% of AEV
1	17.34	17.34
2	16.86	34.20
3	12.58	46.78

where *PC* principal component, *EV* explained variance, and *AEV* accumulated explained variance

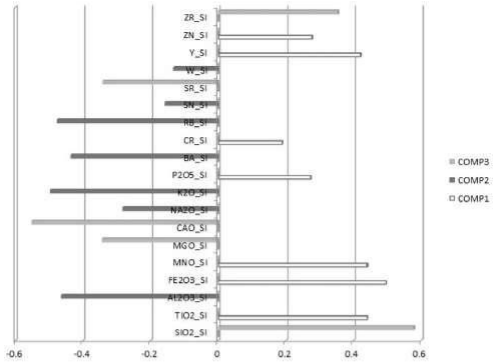

Fig. 3 Disjoint components obtained by the CBPSO-PCA algorithm for environmental data

iterations, reaching its best fit that differs by 0.051 from the fit achieved when using the usual PCA algorithm, indicating a lost in fit but a gain in interpretation; see Fig. 4. The variables to select, according to our proposal, are the variables that appear in the first disjoint principal component, which is the one with the largest percentage of explained variance. There are seven variables with non-zero loadings, which are:

- TIO2_SI,
- FE2O3_SI,
- MNO_SI,
- P2O5_SI,
- CR_SI,
- Y_SI,
- ZN_SI.

The ratio between the number of original variables and the number of selected variables is $19/7 = 2.71$, which satisfies the recommendation given in Grossman et al. (1991).

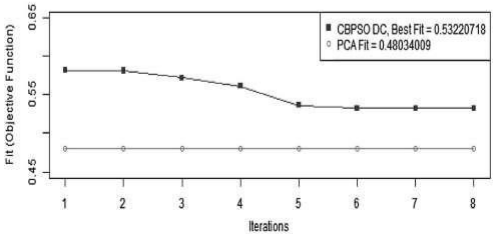

Fig. 4 Convergence of the CBPSO-PCA algorithm with environmental data according to the method indicated

5 Conclusions, discussion and future research

This paper reported the following findings:

(i) A new methodology for calculating principal components based on an intelligent search is proposed, which uses binary optimization with constrains by means of particle swarm.
(ii) A stochastic optimization method was used to find solutions in cases of high computational complexity.
(iii) A numerical evaluation of the proposed methodology was considered by means of Monte Carlo simulations.
(iv) By using a case study with real-world environmental data, we have illustrated the new methodology for DPCA.

The new methodology consists of a PCA with constraints, allowing us to determine components that are linear combinations of disjoint subsets of the original variables. Because the components are of norm one and disjoint, the matrix of factorial loads is orthogonal, enabling us to interpret it as a rotation of the space of individuals. This permits us to preserve the original variability in the rotated space, and to use, in a natural way, the variance of the new variables as an indicator of the variance explained by the disjoint components. By replacing the local search with the intelligent search, the set of feasible solutions is better explored, enabling us high-quality solutions to be reached regardless of the size of the matrix. In addition, the particles have shared memory enabling them to ignore routes that lead to solutions that represent local optima, something which is seen in the percentage of success in obtaining the best solution for the empirical illustration with simulated data. The numerical evaluations of the proposed methodology with simulated and real data sets allowed us to show its good performance and its potential applications. We obtained a new technique which can be a useful knowledge addition to the tool-kit of diverse practitioners, environmental engineers, applied statisticians, and data scientists.

Some open problems that arose from this study are the following:

(i) DPCA seeks the global optimum through successive local optimal choices, that is, it is an algorithm of the type known as greedy algorithms. For small instances, a greedy algorithm usually provides the optimal solution, but in larger instances, it is usually trapped in a local optimum. In general, greedy algorithms provide a good solution, but the quality of this solution usually degrades as the size of the data set increases. This

is shown in the two empirical illustrations with simulated data in Sect. 4. When the number of individuals is small, as in the first simulation (100 individuals and 8 variables), both the DPCA algorithm (that uses alternating least squares) and the proposed algorithm (CBPSO-PCA) reach the best solution at 100% of the simulated cases. Note that by increasing the size of the $\boldsymbol{X}$ matrix (200 individuals and 200 variables), the DPCA algorithm achieves the best solution in 93% of the cases, while the CBPSO-PCA algorithm does so in 100% of the cases. It should be mentioned that the fit obtained with the usual PCA is slightly better than the fit obtained with the DPCA. However, what is lost in fit is compensated by obtaining a factor structure that is easier to interpret. This is an aspect to be improved for the new methodology.

(ii) Regression modeling, errors-in-variables, functional data analysis and PLS regression, based on the proposed methodology, are also of interest (Huerta et al. 2019; Martinez et al. 2019; Carrasco et al. 2020).

(iii) In the present work, the global topology has been used (all particles communicate with all). An open area of research on this topic is to determine other topologies to determine if there are any that accelerate the convergence of the algorithm.

(iv) Inertia decreases in a linear way. Then, it is possible to test our methodology with another type of variation of the coefficients of inertia, such as slowed exponential decrease. In addition, we can try using other functions instead of the sigmoidal function, for example, we can use the probit function.

(v) Other applications of the algorithm developed in the context of multivariate analysis are: discriminant analysis, correspondence analysis, and cluster analysis, as well as the already mentioned functional data analysis and PLS.

(vi) There is also a promising field of applications in the so-called statistical learning; for example, for image compression.

(vii) Another frequently studied problem in environmental sciences is to identify the spatio-temporal variability in environmental fields by using the PCA. A good example is given by Barnston and Livezey (1987) for atmospheric circulation patterns. Nevertheless, Hsieh (2004) stated that the drawback of linear methods is that only linear structures can be correctly extracted from the data, but some environmental data can be highly nonlinear. An impact of nonlinearity in time series of environmental data on the performance of the method proposed in the present study is of interest and deserves further research.

Therefore, the new methodology proposed in this study promotes new challenges and opens issues to be explored from the theoretical and numerical perspectives. Research on these and other issues are in progress and their findings will be reported in future articles.

Acknowledgements The authors thank the editors and reviewers for their constructive comments on an earlier version of this manuscript. The research was partially supported by ESPOL Polytechnic University, Escuela Superior Politécnica del Litoral, ESPOL, Guayaquil, Ecuador (J.A. Ramirez-Figueroa and C. Martin-Barreiro), and by FONDECYT (grant 1200525) from the National Agency for Research and Development (ANID) of the Chilean government (V. Leiva).

References

Alatas B, Akin E (2008) Rough particle swarm optimization and its applications in data mining. Soft Comput 12:1205–1218

Barnston AG, Livezey RE (1987) Classification, seasonality and persistence of low-frequency atmospheric circulation patterns. Mon Weather Rev 115:1083–1126

Beaton D, Chin Fatt C, Abdi H (2014) An exposition of multivariate analysis with the singular value decomposition in R. Computatl Stat Data Anal 72:176–189

Carrasco JMF, Figueroa-Zuniga JI, Leiva V, Riquelme M, Aykroyd RG (2020) An errors-in-variables model based on the Birnbaum-Saunders and its diagnostics with an application to earthquake data. Stoch Environ Res Risk Assess 34:369–380

Chu W, Gao X, Sorooshian S (2011) Fortify particle swarm optimizer with principal components analysis: a case study in improving bound-handling for optimizing high-dimensional and complex problems. IEEE Congr Evolut Comput 2011:1644–1648

Esmin A, Matwin S(2012) Data clustering using hybrid particle swarm optimization. In: *Proceedings of the 13th international conference on intelligent data engineering and automated learning*, pp. 159–1662

Ferrara C, Martella F, Vichi M (2016) Dimensions of well-being and their statistical measurements. Studies in theoretical and applied statistics. Springer, NY, pp 85–99

Freitas A, Macedo E, Vichi M (2020) An empirical comparison of two approaches for CDPCA in high-dimensional data. *Statistical Methods and Applications*, pages in press available at https://doi.org/10.1007/s10260-020-00546-2

Frutos E, Galindo MP, Leiva V (2014) An interactive biplot implementation in R for modeling genotype-by-environment interaction. Stoch Environ Res Risk Assess 28:1629–1641

Gajawada S, Toshniwal D (2012) Projected clustering using particle swarm optimization. Proc Technol 4:360–364

Grossman GD, Nickerson DM, Freeman MC (1991) Principal component analyses of assemblage structure data: utility of tests based on eigenvalues. Ecology 72:341–347

Hastie T, Tibshirani R, Friedman J (2009) The elements of statistical learning. Springer, NY

Hsieh WW (2004) Nonlinear multivariate and time series analysis by neural network methods. Rev Geophy 42(RG1003):1–25

Huerta M, Leiva V, Liu S, Rodriguez M, Villegas D (2019) On a partial least squares regression model for asymmetric data with a chemical application in mining. Chemom Intell Lab Syst 190:55–68

Imran M, Hashim R, Khalid NEA (2013) An overview of particle swarm optimization variants. Proc Eng 53:491–496

Jolliffe IT (1973) Discarding variables in a principal component analysis. II: real data. J R Stat Soc C 22:21–31

Jolliffe IT (2002) Principal component analysis. Springer, New York

King JR, Jackson DA (1999) Variable selection in large environmental data sets using principal components analysis. Environmetrics 10:67–77

Lou S, Wu P, Guo L, Duan Y, Zhang X, Gao J (2020) Sparse principal component analysis using particle swarm optimization. J Chem Eng Jpn 53:327–336

Ma B, Ji H (2012) Particle swarm optimization algorithm establish the model of tobacco ingredients in near infrared spectroscopy quantitative analysis. Int Conf Comput Comput Technol Agric 393:92–98

Macedo E, Freitas A (2015) The alternating least-squares algorithm for CDPCA. In: Plakhov A, Tchemisova T, Freitas A (eds) Optimization in the natural sciences. Springer, NY, pp 173–191

Mahoney MW, Drineas P (2009) CUR matrix decompositions for improved data analysis. Proc Natl Acad Sci U. S. Am 106:697–702

Martí L, García J, Berlanga A, Molina JM (2009) An approach to stopping criteria for multi-objective optimization evolutionary algorithms: The mgbm criterion. In: *2009 IEEE congress on evolutionary computation*, pp. 1263–1270

Martinez S, Giraldo R, Leiva V (2019) Birnbaum-Saunders functional regression models for spatial data. Stoch Environ Res Risk Assess 30:1765–1780

Nezamabadi-Pour H, Rostami-Sharbabaki M, Farsangi M (2008) Binary particle swarm optimization: challenges and new solutions. J Comput Soc Iran 6:21–32

Nieto-Librero AB (2015) *Inferential version of biplot methods based on bootstrap resampling and its application to three-way tables.*. PhD thesis, Universidad de Salamanca, Salamanca, Spain (In Spanish)

Nieto-Librero AB, Sierra-Fernández C, Vicente-Galindo MP, Ruíz-Barzola O, Galindo-Villardón MP (2017) Clustering disjoint hj-biplot: a new tool for identifying pollution patterns in geochemical studies. Chemosphere 176:389–396

R Core Team (2018) R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria

Sangwook L, Sangmoon S, Sanghoun O, Witold P, Moongu J (2008) Modified binary particle swarm optimization. Prog Natl Sci 18:1161–1166

Song S, Wang Q, Chen J, Li Y, Zhang W, Ruan Y (2017) Fuzzy c-means clustering analysis based on quantum particle swarm optimization algorithm for the grouping of rock discontinuity sets. J Civ Eng 21:1115–1122

Sorzano COS, Vargas J, Montano AP (2014) A survey of dimensionality reduction techniques. Retrieved September 5, 2020, from http://arxiv.org/abs/1403.2877

Van der Merwe DW, Engelbrecht A (2003) Data clustering using particle swarm optimization. In: *The 2003 congress on evolutionary computation, 2003*, vol. 1, pp. 215–220

Vasile CI, Buiu C (2011) A software system for collaborative robotics applications and its application in particle swarm optimization implementations. Appl Soft Comput J 11:5498–5507

Vichi M, Saporta G (2009) Clustering and disjoint principal component analysis. Comput Stat Data Anal 53:3194–3208

Vigneau E, Qannari EM (2003) Clustering of variables around latent components. Commun Stat Simul Comput 32:1131–1150

Vines SK (2000) Simple principal components. J R Stat Soc C 49:441–451

Voss MS (2005) Principal component particle swarm optimization: a step towards topological swarm intelligence. In: *2005 IEEE congress on evolutionary computation*, vol. 1, pp. 298–305

Wang L, Liu X, Sun M, Qu J, Wei Y (2018) A new chaotic starling particle swarm optimization algorithm for clustering problems. Math Probl Eng 2018:1–14

Zhao X, Lin W, Zhang Q (2014) Enhanced particle swarm optimization based on principal component analysis and line search. Appl Math Comput 229:440–456

Zou H, Hastie T, Tibshirani R (2006) Sparse principal component analysis. J Comput Gr Stat 15:265–286

Bibliografía:

Alatas, B., & Akin, E. (2008). Rough particle swarm optimization and its applications in data mining. *Soft Computing*, *12*(12), 1205–1218. https://doi.org/10.1007/s00500-008-0284-1

Almorox-González, P., González-Partida, J. T., Burgos-García, M., De La Morena-Álvarez-Palencia, C., Arche-Andradas, L., & Dorta-Naranjo, B. P. (2007). Portable high resolution LFM-CW radar sensor in millimeter-wave band. *2007 International Conference on Sensor Technologies and Applications, SENSORCOMM 2007, Proceedings*, *21*(4), 5–9. https://doi.org/10.1007/s12205-016-1223-9

Beale, E. M., Kendall, M. G., & Mann, D. W. (1967). The discarding of variables in multivariate analysis. *Biometrika*, *54*(3), 357–366. https://doi.org/10.1093/biomet/54.3-4.357

Beaton, D., Chin Fatt, C., & Abdi, H. (2014). An Exposition of multivariate analysis with the singular value decomposition in R. *Computational Statistics and Data Analysis*, *72*, 176–189. https://doi.org/10.1016/j.csda.2013.11.006

Bibo M., & Haiyan J. (2012). Particle Swarm Optimization Algorithm Establish the Model of Tobacco Ingredients in Near Infrared Spectroscopy Quantitative Analysis. *International Conference on Computer and Computing Technologies in Agriculture*, *IFIPAICT,*(Conference paper), pp 92-98. about:blank

Camacho, J., Pérez-Villegas, A., Rodríguez-Gómez, R., & Jiménez-Mañas,

E. (2015). Multivariate Exploratory Data Analysis (MEDA) Toolbox for Matlab. In *Chemometrics and Intelligent Laboratory Systems* (Vol. 143). https://doi.org/10.1016/j.chemolab.2015.02.016

Chu, W., Gao, X., & Sorooshian, S. (2011). Fortify particle swam optimizer (PSO) with principal components analysis: A case study in improving bound-handling for optimizing high-dimensional and complex problems. *2011 IEEE Congress of Evolutionary Computation, CEC 2011*, 1644–1648. https://doi.org/10.1109/CEC.2011.5949812

Cobre, A. de F., Böger, B., Vilhena, R. de O., Fachi, M. M., dos Santos, J. M. M. F., & Tonin, F. S. (2020). A multivariate analysis of risk factors associated with death by Covid-19 in the USA, Italy, Spain, and Germany. *Journal of Public Health (Germany)*, 1–7. https://doi.org/10.1007/s10389-020-01397-7

Cuadras, C. (2014). Nuevos Métodos De Análisis Multivariante. In *CMC Editions*. https://doi.org/10.1017/CBO9781107415324.004

Devkota, J. U. (2021). Multivariate Analysis of COVID-19 for Countries with Limited and Scarce Data: Examples from Nepal. *Journal of Environmental and Public Health*, *2021*. https://doi.org/10.1155/2021/8813505

Di Salvo, F., Ruggieri, M., & Plaia, A. (2015). Functional principal component analysis for multivariate multidimensional environmental data. *Environmental and Ecological Statistics*. https://doi.org/10.1007/s10651-015-0317-8

Eberhart, & Yuhui, S. (2001). Particle swarm optimization: developments, applications and resources. *Evolutionary Computation, 2001. Proceedings of the 2001 Congress On*, *1*(February 2001), 81–86 vol. 1. https://doi.org/10.1109/CEC.2001.934374

Eckart, C., & Young, G. (1936). The approximation of one matrix by another of lower rank. *Psychometrika*, *1*(3), 211–218. https://doi.org/10.1007/BF02288367

Embling, C. B., Illian, J., Armstrong, E., van der Kooij, J., Sharples, J., Camphuysen, K. C. J., & Scott, B. E. (2012). Investigating fine-scale spatio-temporal predator-prey patterns in dynamic marine ecosystems: A functional data analysis approach. *Journal of Applied Ecology*. https://doi.org/10.1111/j.1365-2664.2012.02114.x

Eriksson, L., Byrne, T., Johansson, E., Trygg, J., Vikström, C., & Wold, S. (2006). *Multi-and Megavariate Data Analysis Basic Principles and Applications*.

Esmin, A. A. A., & Matwin, S. (2012). Data clustering using hybrid particle swarm optimization. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Vol. 7435 LNCS* (pp. 159–166). Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-32639-4_20

Febrero-Bande, M., & Oviedo de la Fuente, M. (2011). Refrence manual of R package: Functional Data Analysis and Utilities for Statistical Computing (fda.usc). In *The Comprehensive R Archive Network (CRAN)*.

Ferrara, C., Martella, F., & Vichi, M. (2016). Dimensions of Well-Being and Their Statistical Measurements. In *Topics in Theoretical and Applied Statistics* (pp. 85–99). Springer International Publishing. https://doi.org/10.1007/978-3-319-27274-0_8

Ferrara, C., Martella, F., & Vichi, M. (2018). *Probabilistic Disjoint Principal Component Analysis*. https://iris.uniroma1.it/handle/11573/1112999

Ferreira, M. L. (2020). A multivariate analysis on spatiotemporal evolution of Covid-19 in Brazil. *Infectious Disease Modelling*, *5*, 670–680. https://doi.org/10.1016/j.idm.2020.08.012

Gabriel, K. R. (1971). The biplot graphic display of matrices with application to principal component analysis. *Biometrika*, *58*(3), 453–467. https://doi.org/10.1093/biomet/58.3.453

Gajawada, S., & Toshniwal, D. (2012). Projected Clustering Using Particle Swarm Optimization. *Procedia Technology*, *4*, 360–364. https://doi.org/10.1016/j.protcy.2012.05.055

Galindo, M. P. (1986). Una alternativa de representación simultánea: HJ-Biplot. In *Qüestió* (Vol. 10, pp. 13–23).

Grossman, G. D., Nickerson, D. M., & Freeman, M. C. (1991). Principal component analyses of assemblage structure data: utility of tests based on eigenvalues. *Ecology*. https://doi.org/10.2307/1938927

Hadjipantelis, P. Z., & Müller, H.-G. (2018). *Functional Data Analysis for Big Data: A Case Study on California Temperature Trends*. 457–483.

https://doi.org/10.1007/978-3-319-18284-1_18

Hastie T, Tibshirani R, & Friedman J. (2017). *The Elements of Statistical Learning*. Springer New York. https://doi.org/10.1007/b94608

Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, *24*(6), 417–441. https://doi.org/10.1037/h0071325

Hsing, T., & Eubank, R. (2013). Theoretical Foundations of Functional Data Analysis, with an Introduction to Linear Operators. In *Theoretical Foundations of Functional Data Analysis, with an Introduction to Linear Operators*. John Wiley & Sons, Ltd. https://doi.org/10.1002/9781118762547

Imran, M., Hashim, R., & Khalid, N. (2013). An overview of particle swarm optimization variants. *Procedia Engineering*, *53*(1), 491–496. https://doi.org/10.1016/j.proeng.2013.02.063

Jolliffe, I. (2002). Principal Component Analysis, Second Edition. *Encyclopedia of Statistics in Behavioral Science*, *30*(3), 487. https://doi.org/10.2307/1270093

Jolliffe, I. T. (1972). Discarding Variables in a Principal Component Analysis. I: Artificial Data. *Applied Statistics*, *21*(2), 160. https://doi.org/10.2307/2346488

Jolliffe, I. T. (1973). Discarding Variables in a Principal Component Analysis. II: Real Data. *Applied Statistics*. https://doi.org/10.2307/2346300

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. *Neural Networks, 1995. Proceedings., IEEE International Conference On*, *4*, 1942–1948 vol.4. https://doi.org/10.1109/ICNN.1995.488968

King, J. R., & Jackson, D. A. (1999). Variable selection in large environmental data sets using principal components analysis. *Environmetrics*. https://doi.org/10.1002/(sici)1099-095x(199901/02)10:1<67::aid-env336>3.0.co;2-0

Kokoska, P., & Reimherr, M. (2017). Introduction to Functional Data Analysis. In T. & F. Group (Ed.), *Introduction to Functional Data Analysis* (First). https://doi.org/10.1007/978-0-387-98185-7_1

Konishi, T. (2020). Principal component analysis of coronaviruses reveals their diversity and seasonal and pandemic potential. *PLOS ONE*, *15*(12), e0242954. https://doi.org/10.1371/journal.pone.0242954

Kumar, A., Rani, P., Kumar, R., Sharma, V., & Purohit, S. R. (2020). Data-driven modelling and prediction of COVID-19 infection in India and correlation analysis of the virus transmission with socio-economic factors. *Diabetes and Metabolic Syndrome: Clinical Research and Reviews*, *14*(5), 1231–1240. https://doi.org/10.1016/j.dsx.2020.07.008

Leng, X., & Müller, H. G. (2006). Classification using functional data analysis for temporal gene expression data. *Bioinformatics*. https://doi.org/10.1093/bioinformatics/bti742

Lu, Z. Q. J. (2007). Nonparametric Functional Data Analysis: Theory And Practice. *Technometrics*. https://doi.org/10.1198/tech.2007.s483

Luo, Y., Wu, J., Lu, J., Xu, X., Long, W., Yan, G., Tang, M., Zou, L., Xu, D., Zhuo, P., Si, Q., & Zheng, X. (2020). Investigation of COVID-19-related symptoms based on factor analysis. *Annals of Palliative Medicine*, *9*(4), 1851–1858. https://doi.org/10.21037/apm-20-1113

Macedo, E., & Freitas, A. (2015). The alternating least-squares algorithm for CDPCA. *Communications in Computer and Information Science*, *499*, 173–191. https://doi.org/10.1007/978-3-319-20352-2_12

MacGregor, J. F., Bruwer, M. J., Miletic, I., Cardin, M., & Liu, Z. (2015). Latent Variable Models and Big Data in the Process Industries. *IFAC-PapersOnLine*, *48*(8), 520–524. https://doi.org/10.1016/j.ifacol.2015.09.020

Mahmoudi, M. R., Heydari, M. H., Qasem, S. N., Mosavi, A., & Band, S. S. (2021). Principal component analysis to study the relations between the spread rates of COVID-19 in high risks countries. *Alexandria Engineering Journal*, *60*(1), 457–464. https://doi.org/10.1016/j.aej.2020.09.013

Mahoney, M. W., & Drineas, P. (2009). CUR matrix decompositions for improved data analysis. *Proceedings of the National Academy of Sciences of the United States of America*, *106*(3), 697–702. https://doi.org/10.1073/pnas.0803205106

Müller, H. G., Sen, R., & Stadtmüller, U. (2011). Functional data analysis for volatility. *Journal of Econometrics*, *165*(2), 233–245. https://doi.org/10.1016/j.jeconom.2011.08.002

Nieto-Librero, A., Sierra-Fernández, C., Vicente-Galindo, M., Ruíz-Barzola, O., & Galindo-Villardón, M. (2017). Clustering Disjoint HJ-Biplot: A new tool for identifying pollution patterns in geochemical studies. *Chemosphere*, *176*, 389–396. https://doi.org/10.1016/j.chemosphere.2017.02.125

Nieto-Librero, Ana. (2015). *Versión inferencial de los métodos biplot basada en remuestreo bootstrap y su aplicación a tablas de tres vías*. Universidad de Salamanca (España).

Pearson, K. (1901). LIII. On lines and planes of closest fit to systems of points in space . *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, *2*(11), 559–572. https://doi.org/10.1080/14786440109462720

Peng, D., Nie, Q., Bi, Y., & Liu, W. (2012). Particle swarm optimization-based wavelet packet regression for multivariate analysis of near-infrared spectroscopy. *2012 5th International Conference on Biomedical Engineering and Informatics, BMEI 2012*, 971–975. https://doi.org/10.1109/BMEI.2012.6513066

Poli, R., Kennedy, J., & Blackwell, T. (2007). Particle swarm optimization An overview. *Swarm Intell*, *1*(1), 33–57. https://doi.org/10.1109/ICNN.1995.488968

Pourshoghi, A., Zakeri, I., & Pourrezaei, K. (2016). Application of functional data analysis in classification and clustering of functional near-infrared spectroscopy signal in response to noxious stimuli. *Journal of Biomedical Optics*.

https://doi.org/10.1117/1.jbo.21.10.101411

Ramsay, J. O., & Silverman, B. W. (2005). Functional data analysis, 2nd edition. In *Springer series in statistics*. https://doi.org/10.1007/b98888

Ramsay, J. O., Wickham, H., Graves, S., & Hooker, G. (2014). Package "fda": Functional data analysis. In *CRAN*.

Ramsay, J., & Ramsey, J. (2002). Functional data analysis of the dynamics of the monthly index of nondurable goods production. *Journal of Econometrics*. https://doi.org/10.1016/S0304-4076(01)00127-0

Romeu, J. (2020a). *More on Applying Principal Components & Discrimination Analysis to Covid-19*. https://doi.org/10.13140/RG.2.2.31552.35840

Romeu, J. (2020b). *Multivariate Stats (PC & Discrimination) in the Analysis of Covid-19*. https://doi.org/10.13140/RG.2.2.30529.17766

Sanchez-Garcia, M., Extremera, N., & Fernandez-Berrocal, P. (2016). The factor structure and psychometric properties of the Spanish version of the Mayer-Salovey-Caruso Emotional Intelligence Test. *Psychological Assessment*, *28*(11), 1404–1415. https://doi.org/10.1037/pas0000269

Scholz, U., Gutiérrez, B., Sud, S., & Schwarzer, R. (2002). Is General Self-Efficacy a Universal Construct? *European Journal of Psychological Assessment*, *18*(3), 242–251. https://doi.org/https://doi.org/10.1027//1015-5759.18.3.242

Schwarzer, R;Jerusalem, M. (1995). Generalized Self-Efficacy scale.

Measures in Health Psychology: A User's Portfolio. Causal and Control Beliefs, 35–37. https://doi.org/10.1027//1015-5759.18.3.242.

Sørensen, H., Goldsmith, J., & Sangalli, L. M. (2013). An introduction with medical applications to functional data analysis. *Statistics in Medicine*. https://doi.org/10.1002/sim.5989

Sorzano, C. O. S., Vargas, J., & Montano, A. P. (2014, March 12). *A survey of dimensionality reduction techniques*. http://arxiv.org/abs/1403.2877

Su, X., Yan, X., & Tsai, C. L. (2012). Linear regression. *Wiley Interdisciplinary Reviews: Computational Statistics*. https://doi.org/10.1002/wics.1198

Valderrama, M., Ocaña, F., & Aguilera, A. M. (2000). *Predicción dinámica mediante análisis de datos funcionales*. https://www.abebooks.com/Predicción-dinámica-mediante-análisis-datos-funcionales/19183842399/bd

Van Der Merwe, D. W., & Engelbrecht, A. P. (2003). Data clustering using particle swarm optimization. *2003 Congress on Evolutionary Computation, CEC 2003 - Proceedings*, *1*, 215–220. https://doi.org/10.1109/CEC.2003.1299577

Vasile, C. I., & Buiu, C. (2011). A software system for collaborative robotics applications and its application in particle swarm optimization implementations. *Applied Soft Computing Journal*, *11*(8), 5498–5507. https://doi.org/10.1016/j.asoc.2011.05.009

Vega-Hernández, M. C., Patino-Alonso, M. C., Cabello, R., Galindo-

Villardón, M. P., & Fernández-Berrocal, P. (2017). Perceived emotional intelligence and learning strategies in Spanish University students: A new perspective from a Canonical non-symmetrical correspondence analysis. *Frontiers in Psychology*, *8*(OCT), 1–10. https://doi.org/10.3389/fpsyg.2017.01888

Vichi, M., & Saporta, G. (2009). Clustering and disjoint principal component analysis. *Computational Statistics and Data Analysis*, *53*(8), 3194–3208. https://doi.org/10.1016/j.csda.2008.05.028

Vigneau, E., & Qannari, E. (2003). Clustering of Variables Around Latent Components. *Communications in Statistics Part B: Simulation and Computation*, *32*(4), 1131–1150. https://doi.org/10.1081/SAC-120023882

Villegas, G., González, N., Sánchez-García, A. B., Sánchez, M., & Galindo-Villardón, M. P. (2018). Seven methods to determine the dimensionality of tests: Application to the general self-efficacy scale in twenty-six countries. *Psicothema*, *30*(4), 442–448. https://doi.org/10.7334/psicothema2018.113

Vines, S. (2000). Simple Principal Components. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, *49*(4), 441–451.

Viviani, R., Grön, G., & Spitzer, M. (2005). Functional principal component analysis of fMRI data. *Human Brain Mapping*. https://doi.org/10.1002/hbm.20074

Voss, M. (2005). Principal component particle swarm optimization: a step

towards topological swarm intelligence. *2005 IEEE Congress on Evolutionary Computation*, *1*. https://doi.org/10.1109/CEC.2005.1554698

Wang, L., Liu, X., Sun, M., Qu, J., & Wei, Y. (2018). A New Chaotic Starling Particle Swarm Optimization Algorithm for Clustering Problems. *Mathematical Problems in Engineering*, *2018*, 1–14. https://doi.org/10.1155/2018/8250480

Yanai, H., Takeuchi, K., & Takane, Y. (2011). Projection Matrices, Generalized Inverse Matrices, and Singular Value Decomposition. In *Projection Matrices, Generalized Inverse Matrices, and Singular Value Decomposition* (First). Springer-Verlag, NewYork. https://doi.org/10.1007/978-1-4419-9887-3

Ye, W., Lu, W., Tang, Y., Chen, G., Li, X., Ji, C., Hou, M., Zeng, G., Lan, X., Wang, Y., Deng, X., Cai, Y., Huang, H., & Yang, L. (2020). Identification of COVID-19 Clinical Phenotypes by Principal Component Analysis-Based Cluster Analysis. *Frontiers in Medicine*, *7*, 570614. https://doi.org/10.3389/fmed.2020.570614

Zhao, X., Lin, W., & Zhang, Q. (2014). Enhanced particle swarm optimization based on principal component analysis and line search. *Applied Mathematics and Computation*, *229*, 440–456. https://doi.org/10.1016/j.amc.2013.12.068

Zou, H., Hastie, T., & Tibshirani, R. (2004). Sparse principal component analysis. *Journal of Computational and Graphical Statistics*, *15*(2), 262–286. https://doi.org/10.1198/106186006X113430

ANEXOS

Código en R para realizar el cálculo de las componentes principales funcionales clásicas. por medio de la función “pca.fd” de la librería “fda” (Rcran)

En el caso de los datos utilizados tenemos:

```
mounthbasis11 <- create.fourier.basis(c(0, 12), 11, 12, axes=list("axesIntervals"))
```

Una vez que se ha creado la base de Fourier, el siguiente paso consiste en obtener las funciones de aproximación x, que son combinaciones lineales de las funciones de la base que creamos anteriormente:

```
mounthtempfd <- with(CanadianWeather, smooth.basis(monthEnd.5, monthlyTemp,
mounthbasis11)$fd)
```

Obtenemos las componentes principales funcionales por medio de la función “pca.fd” y las asignamos a “mounthtemppcaobj”:

```
mounthtemppcaobj <- pca.fd(mounthtempfd, nharm=5)
```

donde “nharm” es el número de armónicos o componentes principales a graficar
En el ejemplo “nharm=5”

Y luego procedemos a obtener sus gráficos respectivos

```
plot.pca.fd(mounthtemppcaobj, cex.main=0.9)
```