



Tesis Doctoral en Ingeniería Informática

# Resolución de problemas de detección y clasificación mediante soluciones óptimas no supervisadas

Angel Ramiro Mur Güerri

Máster en Ingeniería de Sistemas y de Control

Directora: Dra. Raquel Dormido Canto

Co-Directora: Dra. Natividad Duro Carralero

Departamento de Informática y Automática  
Escuela Técnica Superior de Ingeniería Informática  
Universidad Nacional de Educación a Distancia

Madrid, 2017



Tesis Doctoral en Ingeniería Informática

# Resolución de problemas de detección y clasificación mediante soluciones óptimas no supervisadas

Angel Ramiro Mur Güerri

Máster en Ingeniería de Sistemas y de Control

Directora: Dra. Raquel Dormido Canto

Co-Directora: Dra. Natividad Duro Carralero

Departamento de Informática y Automática  
Escuela Técnica Superior de Ingeniería Informática  
Universidad Nacional de Educación a Distancia

Madrid, 2017



## **Agradecimientos**

Quiero expresar mi más sincero agradecimiento a todas las personas e instituciones, que en mayor o menor medida, han contribuido a la realización de esta Tesis Doctoral.

En particular:

A la Dra. Raquel Dormido Canto y la Dra. Natividad Duro Carralero por haberme animado a la realización de la tesis, sus orientaciones, consejos y apoyo en todo momento.

Al Dr. Jesús Vega Sánchez y al Dr. Sebastián Dormido Canto por su ayuda, su disponibilidad, sus consejos e interés en mi trabajo.

A todos, muchas gracias

Angel Mur

Madrid, España

Marzo 2017



## Resumen

El problema de la detección de eventos en una señal o serie temporal puede ser definido como un problema de clasificación no supervisada. La detección se realiza sin tener a priori conocimiento alguno sobre la naturaleza de los eventos.

El problema de la clasificación no supervisada de señales puede ser tratado como un problema de clasificación de secuencias temporales. Las señales se clasifican según su comportamiento. Estas pueden tener diferente duración y sus eventos una localización temporal variable.

Ambos problemas, detección y clasificación, están relacionados y, como se aborda en esta tesis, pueden ser resueltos mediante técnicas de agrupamiento de objetos o Clustering. En este trabajo se propone utilizar el Clustering para obtener una solución óptima y no supervisada.

La resolución de ambos problemas puede aplicarse a señales multicanal y resulta de interés en dominios tan variados como la bioingeniería, geofísica, fusión nuclear, etc.

A partir de la detección de eventos en una señal, además de la clasificación de señales, se derivan otras aplicaciones. Por ejemplo, el análisis de los estados de una señal. Se entiende por estado la porción de señal entre dos eventos consecutivos.

Cada estado de una señal puede analizarse mediante el estudio de sus componentes independientes. En esta tesis, se presenta un método para determinar su número óptimo de forma no supervisada.

## Summary

The event detection problem in a signal or temporal series can be defined as an unsupervised classification problem. The detection is performed without any prior knowledge about the nature of the events.

The unsupervised classification problem of signals can be managed as a classification problem of temporal sequences. The signals are classified according to their behavior. These can have different duration and their events an uncertain temporal location.

Both problems, detection and classification, are linked and, as addressed in this thesis, they can be solved through grouping techniques of objects or Clustering. In this work, it is proposed to use Clustering to obtain an unsupervised optimal solution.

The solutions to both problems can be applied to multichannel signals and it is interesting in different fields as diverse as bioengineering, geophysics, nuclear fusion, etc.

Thanks to event detection in a signal, in addition to the signal classification, other applications arise. For example, the analysis of states in a signal. A state is the signal portion between two consecutive events.

Each state of a signal can be analysed by means of its independent components. In this thesis, a method to determine its optimal number in an unsupervised way is presented.







# Índice

	Pág.
Resumen .....	vi
Summary .....	vii
Acrónimos .....	xii
CAPÍTULO 1: INTRODUCCIÓN .....	1
1.1 Objetivos de la tesis .....	4
1.2 Aplicaciones del método de detección .....	5
1.3 Estado del arte de los métodos de detección de eventos .....	6
1.3.1 Métodos de “Control Chart” .....	7
1.3.2 Métodos de detección secuencial .....	11
1.3.3 Métodos paramétricos de optimización .....	14
1.3.4 Métodos no paramétricos .....	15
1.3.5 Métodos de detección mediante identificación .....	17
1.4 Estructura de la memoria de la tesis .....	19
CAPÍTULO 2: MARCO TEÓRICO .....	23
2.1 Clustering y detección de eventos .....	23
2.2 Clustering y clasificación de series temporales .....	28
2.3 Independent Component Analysis y análisis de estados .....	30
CAPÍTULO 3: OBJETIVOS Y RESULTADOS DE CADA PUBLICACIÓN .....	33
3.1 Análisis de <i>ART.1</i> : Unsupervised Event Characterization and Detection in Multichannel Signals: An <i>EEG</i> application .....	34
3.2 Análisis de <i>ART.2</i> : Unsupervised event detection and classification of multichannel signals .....	35
3.3 Mejoras para <i>UMED</i> y <i>UMEDC</i> .....	37
3.4 Análisis de <i>ART.3</i> : Determination of the optimal number of clusters using a spectral clustering optimization .....	39
3.5 Análisis de <i>ART.4</i> : An unsupervised method to determine the optimal number of independent components .....	41

CAPÍTULO 4: <i>ART.1:</i> Unsupervised Event Characterization and Detection in Multichannel Signals: An <i>EEG</i> application .....	45
CAPÍTULO 5: <i>ART.2:</i> Unsupervised event detection and classification of multichannel signals .....	70
CAPÍTULO 6: <i>ART.3:</i> Determination of the optimal number of clusters using a spectral clustering optimization .....	98
CAPÍTULO 7: <i>ART.4:</i> An unsupervised method to determine the optimal number of independent components .....	126
CAPÍTULO 8: CONCLUSIONES .....	146
Bibliografía .....	151
Anexo 1 .....	159
Anexo 2 .....	161

## Acrónimos

<i>AR</i>	→ Coeficiente autoregresivo
<i>ART.1</i>	→ Artículo 1
<i>ART.2</i>	→ Artículo 2
<i>ART.3</i>	→ Artículo 3
<i>ART.4</i>	→ Artículo 4
<i>CCC</i>	→ Coeficiente de correlación Cofenético
<i>CI</i>	→ Componentes Independientes
<i>EEG</i>	→ Señal electroencefalográfica
<i>EMG</i>	→ Señal electromiográfica
<i>GS</i>	→ Método denominado Spectral Global Silhouette
<i>GSWA</i>	→ Método <i>GS</i> para segmentación con <i>WA</i>
<i>GSWB</i>	→ Método <i>GS</i> para segmentación con <i>WB</i>
<i>HC</i>	→ Algoritmo Jerárquico
<i>HMM</i>	→ Modelo Oculto de Markov
<i>ICA</i>	→ Análisis en Componentes Independientes
<i>LCC</i>	→ Correlación Lineal entre Componentes
<i>NO</i>	→ Número óptimo de clusters
<i>NOCI</i>	→ Número óptimo de Componentes Independientes
<i>Nyström_WB</i>	→ Método de segmentación utilizando <i>WB</i> junto al método de Nyström
<i>OC</i>	→ Señal de un solo sensor
<i>SC</i>	→ Clustering o Agrupamiento Espectral
<i>SS</i>	→ Índice Simplificado de Silhouette
<i>MC</i>	→ Señal multicanal
<i>UMED</i>	→ Método no supervisado para la Detección de Eventos
<i>UMEDC</i>	→ Método no supervisado para la Detección de Eventos y la clasificación de señales
<i>WA</i>	→ Método de reducción de datos mediante interpolación y cálculo de <i>NO</i> con <i>GS</i>
<i>WB</i>	→ Método de reducción de datos mediante división en bloques y cálculo de <i>NO</i> con <i>GS</i>



# CAPÍTULO 1

---

## INTRODUCCIÓN

---

Una serie temporal es una señal que representa una secuencia de datos medidos en determinados momentos y ordenados cronológicamente.

En general, una serie temporal proporciona una gran cantidad de información que es necesario interpretar. Una forma de facilitar este análisis consiste en seleccionar la información más significativa. En esta tesis se contribuye a esta selección a partir de la detección de eventos en las series temporales.

Un evento es un cambio en un instante de tiempo determinado  $t_E$ . Se **caracteriza** con la ayuda de un símbolo de un alfabeto particular. De esta forma, un evento de símbolo  $E$  se describe mediante dos elementos  $(E, t_E)$ . Una característica importante es el intervalo de tiempo o la duración  $d_E$  entre dos eventos consecutivos. Con esto, los tres elementos  $(E, t_E, d_E)$  definen un estado. Así, un evento también representa la transición entre dos estados.

**Detectar un evento  $E$**  consiste en determinar el valor de  $t_E$ . Por consiguiente, la detección de un evento es equivalente a detectar el inicio de un nuevo estado.

**La identificación de un evento** se produce cuando el estado  $(E, t_E, d_E)$  se corresponde con un fenómeno conocido. El nombre

específico de dicho fenómeno puede utilizarse para sustituir al símbolo  $E$  del evento.

Un conjunto de eventos en un intervalo de tiempo constituye una **secuencia temporal**. De esta forma, toda serie temporal con eventos se puede representar de forma compacta mediante una secuencia temporal.

Además, dos series temporales con eventos poseen un mismo comportamiento si sus respectivas secuencias temporales son idénticas. Sin embargo, los estados que se caracterizan (o se identifican) con el mismo símbolo (o el mismo fenómeno) pueden tener una duración diferente.

En esta memoria se considera que los términos señal y serie temporal son equivalentes. Sin embargo, se diferencia entre una señal obtenida a partir de un solo sensor ( $OC$ ), de una señal multicanal ( $MC$ ) que representa las señales simultáneas de un conjunto de sensores presentes en un sistema de monitorización.

También se utilizan indistintamente los términos evento y punto de cambio (o simplemente cambio).

Considérese a modo de ejemplo una señal  $MC$  que representa un proceso disruptivo en un reactor de fusión nuclear como la que se muestra en la Figura 1. Esta señal  $MC$  está formada por seis canales. Durante una interrupción, el plasma colapsa de forma incontrolada generando fuerzas mecánicas y calentamientos que amenazan la integridad estructural del dispositivo de fusión.

En este caso, la detección de los eventos permitiría estudiar el proceso disruptivo de forma más simple. Este análisis se hace imprescindible para poder desarrollar técnicas de control que eviten la aparición de la disrupción.

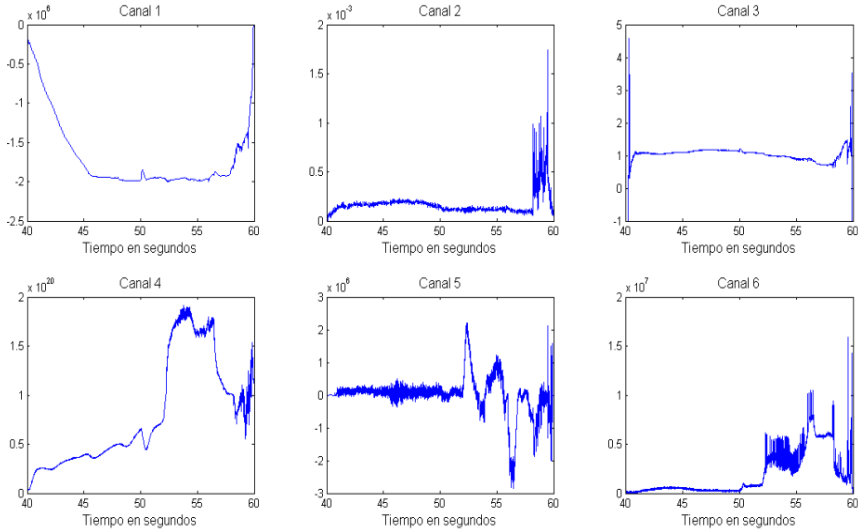


Figura 1: Ejemplo de señal *MC* disruptiva. Detrás de cada canal hay un sensor diferente.

La detección de los eventos del proceso disruptivo por parte de un experto es una tarea difícil ya que la señal es aleatoria, no estacionaria, y en general, se desconoce la naturaleza de los eventos. Por lo tanto, en este tipo de señales es interesante utilizar un método que permita una detección óptima de los eventos de forma no supervisada.



## 1.1 Objetivos de la tesis

En general, los eventos en una señal son desconocidos. En esta tesis se muestra **un nuevo enfoque para detectarlos, caracterizarlos y/o identificarlos**. Además se utiliza este resultado para clasificar señales con eventos y analizar sus estados. Concretamente se pretende:

- 1) **Mostrar un método no supervisado de detección de eventos** con las siguientes características:
  - Debe proporcionar una solución óptima sin que a priori se tenga alguna información sobre la naturaleza de los eventos.
  - Debe poder aplicarse a cualquier tipo de señal (en particular a las *MC*).
  - Debe poder adaptarse para funcionar en tiempo real.
- 2) **Clasificar señales con eventos**. Para este fin, el método además de detectar eventos, debe caracterizarlos y/o identificarlos.

De esta forma, cada señal con eventos se transforma en una secuencia temporal que representa y describe el comportamiento de una señal. Si se tienen varias señales entonces se pueden clasificar de acuerdo a su comportamiento incluso si la duración entre eventos es variable.

- 3) **Analizar los estados de una señal.** Una vez que se han detectado los eventos de una señal también se conocen sus estados. Estos se pueden analizar en su conjunto o también individualmente teniendo en cuenta que:
- El análisis general de los estados implica, por ejemplo, el estudio de su frecuencia, duración, etc.
  - El análisis particular de un estado puede realizarse mediante el estudio de sus componentes independientes. Esto es posible cuando se trata de una señal *MC* (con el mismo tipo de sensores) en donde cada canal representa una proyección de una mezcla de fenómenos independientes. En este caso, es importante la determinación óptima y no supervisada de sus componentes independientes.

## **1.2 Aplicaciones del método de detección**

Una de las características principales del método de detección de eventos propuesto en esta tesis es que puede aplicarse a cualquier señal. En particular, el método puede trabajar con señales *MC* aleatorias no estacionarias. Por ejemplo, las señales en bioingeniería o las interrupciones en fusión nuclear, donde, en general, se desconocen a priori la naturaleza de los eventos. También es práctico utilizarlo en señales donde la tarea de identificación de los eventos por parte de un experto es bastante tediosa.

Las principales aplicaciones son la detección y caracterización de eventos en una señal así como la clasificación de señales. A partir de éstas, se derivan otras aplicaciones como:

- 1) El estudio de secuencias temporales.
- 2) La predicción de eventos disruptivos.
- 3) La detección de anomalías.
- 4) El análisis y filtrado de estados.
- 5) La identificación de eventos.

### **1.3 Estado del arte de los métodos de detección de eventos**

En la literatura existe un gran número de métodos relacionados con la detección de eventos. En esta sección, se describen algunos métodos con el fin de mostrar una visión general de las estrategias de detección más significativas. En concreto se presentan los siguientes:

- 1) Métodos de “Control Chart”.
- 2) Métodos de detección secuencial.
- 3) Métodos paramétricos de optimización.
- 4) Métodos no paramétricos.
- 5) Métodos de detección mediante identificación.

Estos métodos pueden pertenecer a una o varias de las categorías siguientes:

- Métodos paramétricos y no paramétricos: los paramétricos a diferencia de los no paramétricos incorporan información de la distribución de los datos.
- Métodos “offline” y “online”: los métodos “offline” detectan cambios analizando toda la señal de forma estática, mientras que los métodos “online” pueden detectar cambios en tiempo real según se van adquiriendo los datos.
- Métodos *OC* y *MC*: Los *OC* detectan sólo cambios en señales *OC* y los métodos *MC* pueden detectar cambios en señales *MC*.
- Métodos que solo detectan un cambio frente a métodos que detectan varios cambios en una señal.

### 1.3.1 Métodos de “Control Chart”

Los métodos de “Control Chart” [Mon09] determinan un cambio mediante un proceso estadístico. Son métodos paramétricos ya que se aplican sobre datos que siguen una distribución normal. En estos métodos se definen un límite inferior y superior de algún parámetro estadístico relacionado con una señal. Cuando el valor del parámetro sobrepasa los límites se detecta un cambio respecto a lo que es normal. De esta forma, estos métodos son adecuados para detectar los cambios procedentes de una anomalía.

Por ejemplo, el método Cusum [Pag54] utiliza una suma acumulativa de la diferencia entre los valores y el promedio para

detectar un cambio en una señal *OC*. El método “EWMA chart” [Mon09] calcula la media móvil ponderada exponencialmente de las medias anteriores. Ambos métodos también se han adaptado para ser usados en una señal *MC* [YVC03; HME16]. El método denominado "Hotelling's  $T^2$  Control Charts" [CMY99; Eve79] es el más utilizado para señales *MC*.

A continuación se describen mediante ejemplos, el método Cusum y el método Hotelling's  $T^2$ .

### Ejemplo utilizando Cusum

El método Cusum esta diseñado para detectar pequeños cambios respecto a la media. Dada una secuencia  $x_1, x_2, x_3, \dots, x_n$  con media  $m_x$  y desviación estándar  $\sigma_x$ , se define una suma acumulada superior  $U_i$  e inferior  $L_i$  mediante:

$$U_i = \begin{cases} 0 & i=1 \\ \max(0, U_{i-1} + x_i - m_x - K), & i > 1 \end{cases} \quad L_i = \begin{cases} 0 & i=1 \\ \min(0, L_{i-1} + x_i - m_x + K), & i > 1 \end{cases}$$

El parámetro  $K$  es un valor de referencia para regular la detección.

Cusum detecta una anomalía en  $x_j$  si  $U_j > c\sigma_x$  or  $L_j < -c\sigma_x$ .

En el ejemplo de la Figura 2 se ilustra la aplicación de Cusum a una señal *OC* con un valor de  $c=5$ . La gráfica muestra el punto donde la suma acumulada cambia más de cinco desviaciones estándar respecto a la media de referencia.

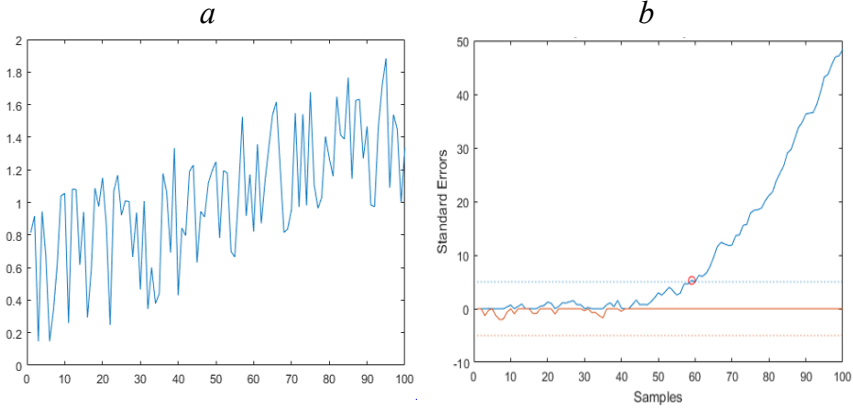


Figura 2: a) Señal con un pequeño cambio. b) Detección del cambio utilizando el método Cusum.

### Ejemplo utilizando Hotelling's $T^2$ Control Charts

Sea  $x_i = (x_{i1}, \dots, x_{ip})'$  la  $i$ -ésima observación de un conjunto de variables aleatorias que siguen una distribución normal multivariable, cuya media es el vector  $\mu$  y la matriz de varianza-covarianza  $\Sigma$ . Si se considera un conjunto de  $n$  observaciones, el vector medio  $m$  del conjunto y la matriz de varianza-covarianza  $S$  del conjunto:

$$m = \begin{bmatrix} m_1 \\ \cdot \\ \cdot \\ m_p \end{bmatrix} \quad S = \frac{1}{n-1} \sum_{i=1}^n (m_i - m)(m_i - m)'$$

pueden utilizarse para estimar  $\mu$  y  $\Sigma$  respectivamente.

## Capítulo 1. Introducción

La estadística Hotelling's  $T^2$  para una observación  $x_i$  es:

$$T^2 = (m_i - m)' S^{-1} (m_i - m)$$

donde  $S^{-1}$  es la inversa de  $S$ . La estadística Hotelling's  $T^2$  mide la distancia estadística de  $x_i$  a  $m$ .

Dado que las variables aleatorias siguen una distribución normal multivariable, el siguiente valor obtenido a partir de la estadística Hotelling's  $T^2$ :

$$\frac{n(n-p)}{p(n+1)(n-1)} T^2$$

sigue una distribución  $F$  con  $p$  y  $n-p$  grados de libertad. Por lo tanto, el valor tabulado de  $F$  para un nivel de significación dado (por ejemplo 0.05), se puede usar como un umbral ( $T^2_{Lim}$ ). Si el valor de la estadística Hotelling's  $T^2$  de una observación  $x_i$  es mayor que el umbral entonces tenemos en  $x_i$  una anomalía.

La Figura 3 muestra un ejemplo en el que Hotelling's  $T^2$  se ha aplicado sobre una señal  $MC$  formada por 3 canales. La representación de  $T^2$  permite visualizar la anomalía detectada.

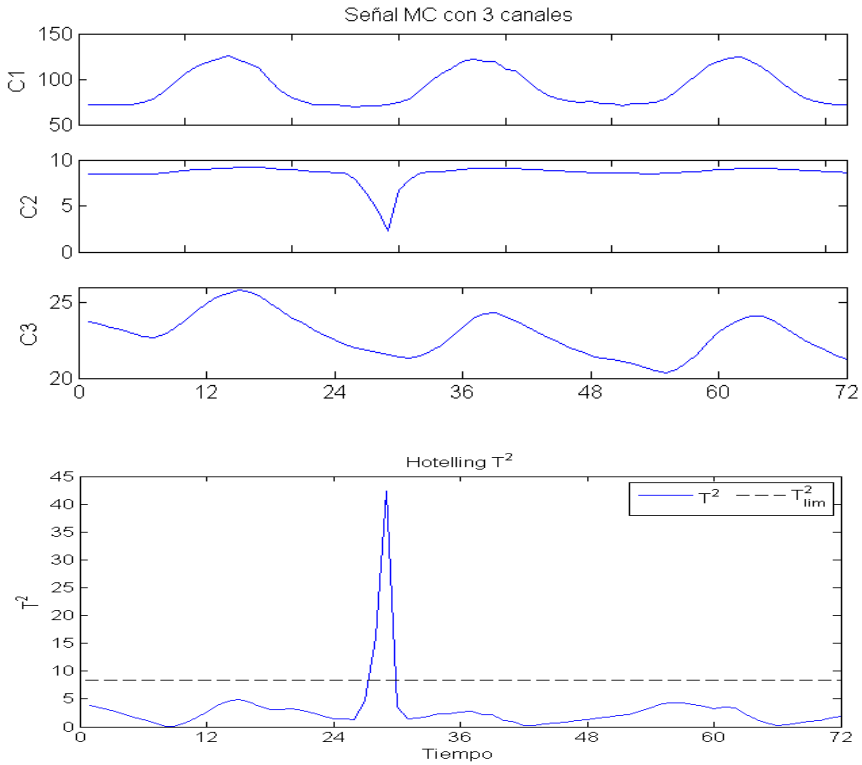


Figura 3: Detección de anomalía utilizando Hotelling  $T^2$  sobre una señal  $MC$  con tres canales.

### 1.3.2 Métodos de detección secuencial

A partir de una secuencia de datos  $A = x_1, x_2, \dots, x_n$  se selecciona la subsecuencia  $B = x_1, \dots, x_w$  en donde  $w < n$ .  $B$  es generada a través de una distribución  $P_0$  conocida a priori.

Son posibles dos situaciones:

- 1) Cada  $x_i$  de  $A$  es generado a partir de  $P_0$ .



- 2) Existe un punto de cambio desconocido  $w < \lambda \leq n$  de forma que los  $x_i$  siguen a  $P_0$  para  $i < \lambda$  y los  $x_i$  siguen otra distribución  $P_1$  para  $w < \lambda \leq i \leq n$ .

Se dice que hay un cambio cuando  $P_1$  difiere de forma significativa de  $P_0$ . Esto ocurre cuando una función que mide la distancia entre las distribuciones  $D_\lambda(P_1, P_0)$  es más grande que un umbral. Esta función depende del punto de cambio  $\lambda$ .

Para cada secuencia  $A$  el algoritmo de detección opera en dos fases:

- primero se localiza el índice  $w < \lambda \leq n$  en donde se obtiene el mayor valor para  $D_\lambda(P_1, P_0)$ ;
- posteriormente se realiza un test de hipótesis para aceptar o rechazar  $\lambda$  como punto de cambio (es decir, se evalúa la hipótesis nula, no hay cambio en  $P_0$ ).

El test de hipótesis compara el mayor  $D_\lambda(P_1, P_0)$  con un umbral. Cuando los datos se adquieren en streaming, este análisis se repite conforme la secuencia  $A$  va cambiando.

Una forma clásica de abordar este problema es utilizando el “Sequential Probability Ratio Test” [BN93; Wal04]. Cuando la distribución cambia de  $P_0$  a  $P_1$  en  $\lambda$ , es normal que la probabilidad de observar la subsecuencia  $C=(x_\lambda, \dots, x_n)$  con  $P_1$  sea significativamente más alta que con  $P_0$ . Es decir, cuando el cociente entre las dos probabilidades sea más alto que un umbral.

Dado que  $P_0$  y  $P_1$  son conocidos a priori y que los datos en streaming son generados independientemente se utiliza como distancia  $D_\lambda(P_1, P_0)$ :

$$D_\lambda(P_1, P_0) = \log \frac{Pr(x_\lambda, \dots, x_n | P_1)}{Pr(x_\lambda, \dots, x_n | P_0)}$$

La mayoría de los algoritmos que comparan distribuciones para detectar cambios utilizan una ventana fija para seleccionar los datos. El resultado depende de la dimensión de la ventana. Por ejemplo en [GGR+02] seleccionan dos ventanas con la misma dimensión y estiman el cambio de distribución entre ellas. También otros autores han propuesto realizar un análisis en paralelo utilizando diversas ventanas de diferente dimensión, pero computacionalmente es costoso [KBG04].

En [Ho05] se propone detectar cambios utilizando la teoría de martingalas y una ventana “pseudo – adaptativa”.

En [MBV+07] se presenta un algoritmo de detección de cambios para datos en streaming, en donde se evitan los problemas de selección de la ventana. No se requiere conocer a priori  $P_1$  y tampoco es necesario que los datos se generen de forma independiente.

### 1.3.3 Métodos paramétricos de optimización

Se han desarrollado métodos "offline" de optimización [KFE12] que buscan encontrar la mejor función escalonada de un cierto parámetro estadístico (por ejemplo la media), que mejor se adapta a un intervalo temporal de una señal  $OC$ . Los distintos cambios en dicha función representan los eventos detectados.

En una secuencia de datos  $x_{1:n} = x_1, x_2, \dots, x_n$ , la solución está formada por  $m$  puntos de cambio ordenados cuyas posiciones son  $\tau_{1:m} = \tau_1, \dots, \tau_m$ . Cada posición es un entero entre 1 y  $n-1$ . Además  $\tau_0 = 0$  y  $\tau_{m+1} = n$ . De esta forma, los  $m$  puntos de cambio dividen los datos en  $m+1$  segmentos. El segmento  $i$ -ésimo contiene los puntos  $x(\tau_{i-1}+1: \tau_i)$ .

La detección de varios puntos de cambio se realiza minimizando:

$$\sum_{i=1}^{m+1} [C(x_{(\tau_{i-1}+1): \tau_i})] + \beta f(m).$$

La función  $C$  es una función de coste para un segmento y  $f(m)$  es un término adicional para corregir el sobreajuste. Por ejemplo, se puede utilizar como función de coste la "log-likelihood" [CG00], y una expresión lineal respecto al número  $m$  para  $\beta$ ,  $f(m) = \beta m$  en donde el parámetro  $\beta$  se justifica mediante el "Akaike's Information Criterion" [Aka74].

Algunos algoritmos de optimización son los siguientes: El “Binary Segmentation” [SK74], el “Segment Neighbourhood method” [AL89] y el “Pruned Exact Linear Time” [KFE12].

La Figura 4 presenta un ejemplo del tipo de solución encontrada utilizando estos algoritmos. Se muestra la función escalonada que mejor se adapta a los cambios de la señal respecto a la media.

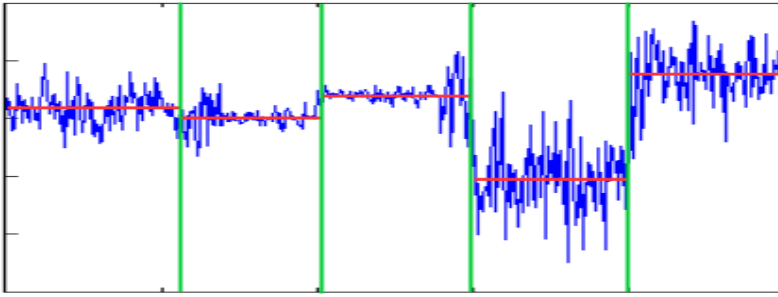


Figura 4: Función escalonada (en rojo) respecto a la media de la señal utilizando un algoritmo de optimización.

### 1.3.4 Métodos no paramétricos

En el mundo real hay muchos procesos cuyo comportamiento no está bien definido y por lo tanto es difícil justificar el uso de modelos paramétricos para detectar cambios.

Sea una secuencia de datos  $x_{1:n} = x_1, x_2, \dots, x_n$ , formada por  $m$  puntos de cambio ordenados cuyas posiciones son  $\tau_{1:m} = \tau_1, \dots, \tau_m$ .

Cada posición es un entero entre 1 y  $n-1$ . Además  $\tau_0 = 0$  y  $\tau_{m+1} = n$ .

De esta forma, los  $m$  puntos de cambio dividen los datos en  $m+1$  segmentos. El segmento  $i$ -ésimo contiene los puntos  $x(\tau_{i-1}+1: \tau_i)$ .

Cada segmento se considera independiente de los demás segmentos e idénticamente distribuido. Si el segmento  $i$ -ésimo tiene como distribución  $P_i$ , la distribución de la secuencia puede escribirse de la forma:

$$PT_x = \left\{ \begin{array}{lll} P_0 & si & i \leq \tau_1; \\ P_1 & si & \tau_1 \leq i \leq \tau_2; \\ P_2 & si & \tau_2 \leq i \leq \tau_3; \\ \dots & & \\ P_m & si & i > \tau_m; \end{array} \right\}$$

La detección de los puntos de cambio puede realizarse a partir de las distribuciones (ver sección 1.3.2). Sin embargo, si en una detección secuencial se utiliza una forma de distribución de datos equivocada aparecen numerosos falsos positivos [RTA11].

El caso más simple es la detección de un único punto de cambio. La detección se resuelve comprobando el siguiente test:

$$H_0: P_0 \quad \forall i, \quad H_1: \left\{ \begin{array}{lll} P_0 & si & i < \tau; \\ P_1 & si & i \geq \tau; \end{array} \right\}$$

donde  $n$  es la longitud de la secuencia y  $\tau < n$ . Como  $\tau$  es desconocido, [Pet79] propuso utilizar un test estadístico  $D_{\tau,n}$  para todos los valores  $1 < \tau < n$  y seleccionar el valor máximo dado por:

$$D_n = \left| \max_{\tau} \frac{D_{\tau,n} - \mu D_{\tau,n}}{\sigma D_{\tau,n}} \right|$$

donde  $\mu$  y  $\sigma$  son la media y desviación estándar correspondientes. La hipótesis nula en donde no hay cambio, se rechaza si  $D_n$  es mayor que un umbral  $h_n$ . El punto de cambio se corresponde con el lugar  $\tau$  en donde  $D_{\tau,n}$  es máximo.

Se utilizan test estadísticos que no dependen de como estén distribuidos los datos. Por ejemplo, el test de Kolgormorov-Smirnov [RA12]. Si el tipo de cambio es conocido (por ejemplo respecto a la media) entonces existen test estadísticos mejor adaptados. Por ejemplo, en [Pet79] se propone utilizar un test estadístico para detectar cambios en la media basado en el Mann-Whitney test [MW47].

El umbral  $h_n$  se elige para limitar la probabilidad de detectar un cambio cuando no ha habido cambio. Se determina a partir de la distribución de  $D_n$ , la cual se estima mediante métodos numéricos [RA12].

### 1.3.5 Métodos de detección mediante identificación

Destacan el “Support Vector Machine” (*SVM*) [LHR13] y los “Hidden Markov Models” (*HMM*) [BRS+07; Ohr01; RR02; Bou09]. Estos métodos utilizan un proceso de aprendizaje. Posteriormente, detectan los eventos identificando a los estados.

## Capítulo 1. Introducción

Pueden trabajar en tiempo real y aplicarse a señales *OC* y *MC*. Sin embargo sólo pueden detectar aquello que han aprendido.

El *SVM* necesita que el aprendizaje sea equilibrado entre los diferentes tipos de estados.

Por otro lado, el *HMM* funciona a partir de algunas suposiciones. Estas impiden que el modelo pueda aplicarse de forma genérica. Por ejemplo, *HMM* supone que una observación es estadísticamente independiente de las observaciones anteriores.

El registro de la actividad eléctrica del cerebro, el electroencefalograma (*EEG*), es una típica señal *MC* con la que se ha trabajado en esta tesis. Un posible evento a detectar en una señal *EEG* es la detección de un guiño (ver Figura 5).

La detección del guiño se ha realizado utilizando un algoritmo *SVM* [LHR13]. Este ha sido entrenado con segmentos de señal *EEG* mezclados con artefactos, cuyo origen está en los movimientos de los ojos. Se observa la posición de los electrodos que han sido proyectados sobre un plano, así como las señales en donde se muestra la detección del guiño.

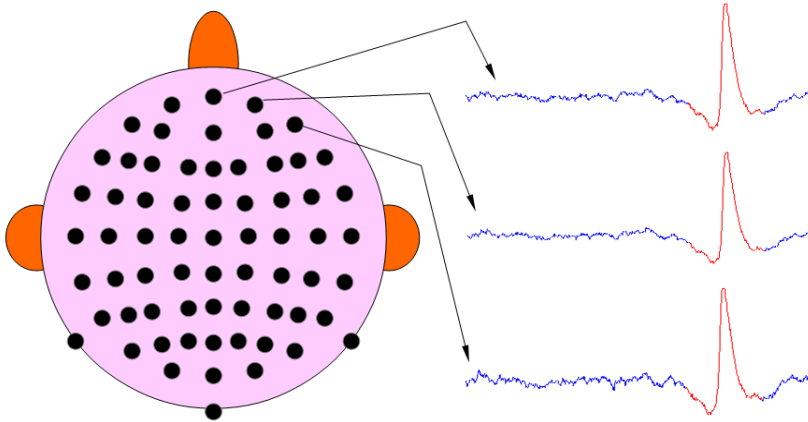


Figura 5: Visualización de la detección de un guiño (en rojo) en tres canales de una señal *EEG*. Se muestra también la posición de los electrodos correspondientes, de las orejas y la nariz.

#### 1.4 Estructura de la memoria de la tesis

Esta memoria está estructurada para presentar una tesis por compendio de publicaciones. Lo más importante del contenido son las diferentes publicaciones, ya que dan respuesta a los objetivos de la tesis. Además de los capítulos con las publicaciones, se han incluido otros capítulos que ayudan a entender el contexto, el marco teórico, los objetivos y los resultados de la tesis.

La memoria de la tesis se estructura de la siguiente forma:

- **Capítulo 1 – Introducción:** En este capítulo se explica el contexto de la tesis, así como la importancia de desarrollar un método de detección de eventos. Se presentan los



## *Capítulo 1. Introducción*

objetivos y las principales aplicaciones que se derivan. También se muestra el estado del arte sobre las diferentes estrategias desarrolladas para la detección de eventos. El capítulo concluye explicando la estructura de la memoria de la tesis.

- **Capítulo 2 – Marco teórico:** Se explica la teoría a partir de la cual se han desarrollado las ideas que permiten dar respuesta a los objetivos de esta tesis. Principalmente, la utilización de Clustering para detectar eventos y clasificar señales. También se ha explicado el análisis de los estados y el uso del “Independent Component Analysis” para el análisis particular de un estado.
- **Capítulo 3 – Objetivos y resultados de cada publicación:** Para cada una de las 4 publicaciones, se describen los principales resultados así como la relación entre el trabajo publicado y los objetivos de la tesis. También se explica la conexión entre las diferentes publicaciones.
- **Capítulo 4:** En este capítulo se incluye el artículo «*ART.1: Unsupervised Event Characterization and Detection in Multichannel Signals: An EEG application*».
- **Capítulo 5:** En este capítulo se incluye el artículo «*ART.2: Unsupervised event detection and classification of multichannel signals*».

## *Capítulo 1. Introducción*

- **Capítulo 6:** En este capítulo se incluye el artículo «*ART.3: Determination of the optimal number of clusters using a spectral clustering optimization*».
- **Capítulo 7:** En este capítulo se incluye el artículo «*ART.4: An unsupervised method to determine the optimal number of independent componentes*».
- **Capítulo 8 – Conclusiones:** En el último capítulo, se exponen las conclusiones que se derivan de los artículos publicados.



## CAPÍTULO 2

---

### MARCO TEÓRICO

---

#### 2.1 Clustering y detección de eventos

Un algoritmo de agrupamiento de objetos, o Clustering, divide un conjunto de objetos en clusters de tal forma que los objetos dentro del mismo cluster son similares, mientras que los objetos en diferentes clusters son distintos [HZ14]. El Clustering es utilizado en numerosos campos como la bioinformática, el text mining, la segmentación de imágenes, etc [JMF99; Jai10; KR05].

El resultado de un algoritmo de Clustering depende de cómo se defina la similaridad entre los objetos. La forma más extendida de expresar la similaridad entre objetos es a través de una distancia en donde una mínima distancia equivale a una máxima similaridad.

Los algoritmos Clustering son variados: Kmeans [Mac67], Expectación Maximización (*EM*) [DLR77], Jerárquico (*HC*) [RM05], Spectral clustering (*SC*) [Lux07; NJW01], etc.

Dado que un Clustering es un método no supervisado, es necesario encontrar una vía para validar la partición de los objetos obtenida. Este proceso de validación es reconocido como esencial para el éxito de las aplicaciones donde el Clustering está presente [JD88; MB02]. Sin embargo, no hay una única medida de validación que pueda considerarse la mejor para cualquier conjunto de objetos.

## *Capítulo 2: Marco teórico*

Como existen varios algoritmos Clustering y distintos tipos de medidas para validar su resultado, en la práctica hay que seleccionar una combinación (algoritmo de Clustering + medida de validación), cuyas propiedades se adapten mejor al tipo de aplicación.

Las medidas de validación pueden clasificarse en dos tipos: medidas de validación exterior y validación interior.

La validación exterior utiliza información exterior no presente en los objetos a agrupar. Se utiliza para evaluar si la estructura encontrada con un algoritmo de Clustering se corresponde con una estructura específica (por ejemplo la proporcionada por un experto). Gracias a la información exterior se conocen el verdadero número de clusters y su contenido. Por lo tanto la validación exterior puede utilizarse para seleccionar qué algoritmo de Clustering es el óptimo para un conjunto específico de objetos.

Algunos de los índices de validación externa más utilizados son: Entropía [SKK00], Pureza [ZK04], F-measure [Van79], Normalized Mutual Information [SG02], el índice de Rand [Ran71], el coeficiente de Jaccard [RV96], el índice de Fowlkes y Mallows [FM83], etc.

En la práctica, la información exterior no está siempre disponible. En tal caso se utiliza la validación interior que es especialmente útil para determinar el algoritmo Clustering más adecuado, así como el óptimo número de clusters.

Como el objetivo principal de un algoritmo de Clustering es agrupar los objetos similares dentro un mismo cluster, y separarlos si no se parecen, en diferentes clusters, las medidas de validación interior se basan en dos criterios [KMP+11; PSK06; ZK02]:

1. **Compacidad:** mide la cercanía de los objetos de un mismo grupo. Un grupo de estas medidas evalúa la compacidad con la varianza. Una varianza pequeña indica una mejor compacidad. Otras medidas estiman la compacidad mediante una distancia: mediante la máxima (o media) distancia entre pares de objetos, o también la máxima distancia (o media) al centro del cluster.
2. **Separación:** mide la diferencia, o como de separado está un cluster de los otros. Por ejemplo, la distancia mínima entre pares de objetos de clusters diferentes, o la distancia entre sus centros son utilizadas para medir la separación. También hay medidas que tienen en cuenta la densidad de los clusters.

En general, el proceso para determinar la mejor partición y número óptimo de clusters de un conjunto de objetos consiste en:

1. Seleccionar una lista de algoritmos de Clustering que serán utilizados sobre el conjunto de objetos.
2. Para cada algoritmo, utilizar diferentes combinaciones de los parámetros (normalmente el número de grupos deseado), para obtener el resultado.

3. Calcular el valor de un índice de validación para cada partición obtenida en 2.
4. Elegir la mejor partición y número óptimo de clusters de acuerdo al valor del índice de validación elegido.

Algunos de los índices de validación interna más utilizados son: El índice Silhouette [Rou87], el índice de Davies–Bouldin [DB79; KR05], el índice de Xie-Beni [KR05; XB91], el índice  $SD$  [HVB00], el índice  $S\_Dbw$  [HV01], el índice de Nearest Neighbors [LLX+13], etc.

**El problema de la detección de eventos en una señal puede ser definido como un problema de clasificación no supervisada utilizando un Clustering.** Esta es la idea principal a partir de la cual se ha desarrollado el método de detección de eventos que se propone en esta tesis. Este método se describe en el artículo *ART.1* (véase la sección 3 del artículo incluido en el capítulo 4 de esta memoria). También se utiliza en el *ART. 2* (véase la sección 3 del artículo incluido en el capítulo 5 de esta memoria).

Cualquier señal  $MC$  puede dividirse en un conjunto de intervalos de pequeñas dimensiones sin solapamiento. Cada intervalo queda caracterizado mediante un vector de características, que contiene una selección de parámetros concatenados de los diferentes canales. Los intervalos pueden obtenerse mediante una ventana deslizándose a lo largo de la señal en donde cada desplazamiento equivale a la anchura de un intervalo (véase la Figura 1 de la sección 3 del

artículo incluido en el capítulo 4 de esta memoria). Los parámetros del vector de características de cada intervalo se corresponden con los obtenidos con la ventana que en su desplazamiento crea el intervalo. El Clustering se aplica utilizando los vectores de características de los intervalos.

El resultado de un algoritmo de Clustering aplicado sobre el conjunto de intervalos depende de forma significativa del vector de características que describe cada intervalo. Los coeficientes autoregresivos (*AR*) de cada ventana son buenos candidatos, ya que son sensibles a los cambios en una señal [LHM+12; DHY95; HPS07; Pec08; Gom08; DLR06]. Además los coeficientes *AR* son invariantes a los cambios de escala de los datos [LHM+12]. De esta forma todos los coeficientes *AR*, que representan a cada intervalo, tienen la misma importancia independientemente de las condiciones de adquisición en cada canal.

Por lo tanto, un Clustering de un conjunto de intervalos (que concatenados forman la señal) los particiona en clusters. En cada cluster, los intervalos están **caracterizados** de la misma forma. Aquellos intervalos consecutivos que pertenecen al mismo cluster forman un **estado**. **Los eventos quedan localizados temporalmente en cada transición entre dos estados**. Mediante un índice de validación interna se consigue determinar el **número óptimo de clusters** (*NO*) y por consiguiente una **óptima detección de los eventos**. Esta optimización se mejora seleccionando la



ventana que mejor capta los coeficientes *AR* a lo largo de la señal. Esta ventana proporciona el mejor valor correspondiente al índice de validación interna seleccionado.

## 2.2 Clustering y Clasificación de Series Temporales

La utilización de un algoritmo de Clustering sobre un conjunto de series temporales es importante en numerosas aplicaciones en las que se requiere encontrar grupos con la misma tendencia o comportamiento. Por ejemplo, en los mercados financieros, la bioingeniería, la geofísica, etc, en donde existen valores que varían en el tiempo como el precio de las acciones, la actividad cerebral, la temperatura, la presión, etc.

La determinación de clusters de series temporales es una tarea compleja, a causa de la dificultad de definir la similaridad. Una vez que la medida de similaridad ha sido definida, se puede utilizar por ejemplo, el método *SC* o el método *HC*, como algoritmos para determinar los grupos. Como todo algoritmo de Clustering, estos proporcionan clusters en donde las series temporales tiene una alta similaridad en cada cluster y una baja similaridad entre series de clusters diferentes. Sin embargo la alta dimensionalidad, el ruido, etc, de las series temporales dificulta conseguir un buen resultado [KGP01; VHG+03].

En un Clustering de series temporales, elegir el tipo de similaridad es fundamental. Sin embargo, no existe un único criterio. Por

ejemplo, una distancia Euclídea refleja una similaridad en tiempo, mientras que la utilización de una “Dynamic Time Warping” [DTG+14] representa una similaridad respecto a la forma.

Por otro lado, la duración variable de las series temporales supone una dificultad adicional para conseguir aplicar con éxito un algoritmo de Clustering.

**El problema de la clasificación no supervisada de series temporales utilizando un algoritmo Clustering, puede ser tratado como un problema de clasificación de secuencias temporales.** Ésta es la idea principal a partir de la cual se ha desarrollado un método de clasificación de series temporales en esta tesis. Este método se describe en el *ART. 2* (véase la sección 3 del artículo incluido en el capítulo 5 de esta memoria).

Primero, este método detecta, caracteriza eventos y transforma cada serie temporal en una secuencia temporal. Los eventos caracterizan el comportamiento de la serie temporal. De esta forma, dos series temporales (independientemente de su duración) tendrán un comportamiento similar, si sus secuencias temporales son parecidas. Posteriormente, un algoritmo de Clustering (por ejemplo el *HC*) puede aplicarse de forma clásica, utilizando las distancias entre todos los pares de secuencias. También se pueden transformar las secuencias temporales en vectores de características con el mismo número de parámetros, y utilizar un algoritmo de Clustering sobre estos vectores de características.

Como se ha mencionado, los coeficientes *AR* (utilizados en la detección y caracterización de los eventos) son invariantes a los cambios de escala. Esto implica que el resultado del algoritmo de Clustering es invariante a las posibles diferencias, en cuanto a la adquisición de las distintas series temporales.

### **2.3 Independent Component Analysis y análisis de estados**

La detección de eventos en una señal permite segmentar la señal en un conjunto de estados consecutivos. Los estados pueden analizarse de forma global. Por ejemplo, estudiando su duración, frecuencia, etc.

Sin embargo, también **se puede realizar un análisis de una parte de la señal** y en particular de un estado, estudiando **cuáles son sus componentes independientes (CIs)**.

Esto es posible cuando se tiene una señal *MC* registrada con el mismo tipo de sensores en todos los canales. Cada canal muestra una señal que representa una proyección de la acción simultánea de diversas fuentes independientes.

Un Independent Component Analysis (*ICA*) sobre una porción de señal proporciona las *CIs*. El *ICA* es una técnica estadística que permite identificar fuentes estadísticamente independientes a partir de una señal *MC* como la descrita en el párrafo anterior. Necesita que el número de sensores sea mayor, o igual, al número de *CIs*.

## Capítulo 2: Marco teórico

Existen varias técnicas *ICA*: FastICA [HO97], *JADE* [CS93], InfoMax [BS95], Mutual Information Least Dependent Component Analysis [SKA+04], etc.

Para analizar un segmento de señal, es interesante estudiar e identificar las diferentes *CI*s. En ocasiones interesa filtrar algunas componentes independientes (relacionadas con artefactos, etc) para dejar el segmento constituido solamente con información útil. En cualquier caso, **la calidad del procesado depende de la determinación del número óptimo de *CI*s (*NOCI*)**. La determinación de *NOCI* mediante un método no supervisado es otro de los objetivos del presente trabajo. Este método se describe en el *ART. 4* (véase el artículo incluido en el capítulo 7 de esta memoria).



## CAPÍTULO 3

---

### OBJETIVOS Y RESULTADOS DE CADA PUBLICACIÓN

---

El método desarrollado sobre detección de eventos se denomina *UMED* (Unsupervised Method for Event Detection). En relación con este método se han publicado dos artículos (*ART.1* y *ART.2*). En el *ART.1* se muestra *UMED* (véase la Sección 3.1 del Capítulo 3). En el *ART.2* se presenta *UMEDC* (Unsupervised Method for Event Detection and Classification of signals) (véase la Sección 3.2 del Capítulo 3).

En la Sección 3.3 de este capítulo se explican otras estrategias a seguir (no mencionadas en *ART.1* y *ART.2* ) para mejorar *UMED* y *UMEDC*.

En la Sección 3.4 se presenta el artículo *ART.3*. En él se explica un nuevo método *GS* (Spectral Global Silhouette) para determinar el número óptimo de clusters (*NO*) a partir de una optimización del algoritmo *SC*.

En la Sección 3.5 se presenta el artículo *ART.4* en el que muestra el método *LCC* (Linear Correlations between Components). *LCC* es un nuevo método no supervisado para la determinación del número óptimo de componentes independientes en una señal.

### **3.1 Análisis de *ART.1: Unsupervised Event Characterization and Detection in Multichannel Signals: An EEG application***

El objetivo principal del artículo *ART.1* es mostrar el método *UMED*. Este artículo presenta *UMED* aplicado a la detección y caracterización de los artefactos en una señal multicanal *EEG*. Una vez los artefactos han sido detectados, es posible analizarlos o suprimirlos.

Otro objetivo del artículo consiste en evaluar el método *UMED* comparando el resultado de la detección de artefactos con la información externa aportada por un método supervisado entrenado para tal fin. A su vez, esta clasificación de los artefactos también está validada manualmente por un experto.

Además de *UMED*, se explican otros 2 métodos *UMED1* y *UMED2*, que se derivan de *UMED*. Estos se utilizan para poder seguir detectando y caracterizando (o identificando) los artefactos cuando la señal *MC* tiene una gran duración. *UMED1* y *UMED2* son también útiles para la detección de artefactos en tiempo real. A diferencia de *UMED1*, que funciona como cualquier otro método supervisado, *UMED2* puede seguir caracterizando nuevos grupos que no hayan sido identificados por un experto.

Una vez que los artefactos han sido detectados, caracterizados e identificados es posible estudiar su frecuencia y duración para realizar estudios psicofisiológicos.

### *Capítulo 3: Objetivos y resultados de cada publicación*

La detección de los artefactos también permite filtrar los artefactos mediante técnicas de *ICA*, sin apenas modificar las zonas de la señal libre de artefactos.

El método *UMED* utiliza el algoritmo jerárquico (*HC*) para el Clustering y el índice de validación interna  $S\_dbw$  para determinar de forma óptima los eventos a lo largo de la señal.

A lo largo de la señal *MC* se desliza una ventana dividiéndola en intervalos, que quedan caracterizados por los coeficientes  $AR(2)$  (coeficientes autoregresivos de orden dos) de cada canal. Los coeficientes  $AR(2)$  para cada intervalo se seleccionan a partir de una ventana óptima. *UMED* posee un criterio matemático (utilizando  $S\_dbw$ ) para la selección de la ventana óptima.

Se ha constatado que los coeficientes  $AR(2)$  son útiles para la detección y caracterización de los artefactos.

No se han empleado técnicas de reducción de componentes en cada intervalo para facilitar que el algoritmo pueda implementarse en tiempo real.

#### **3.2 Análisis de *ART.2*: Unsupervised event detection and classification of multichannel signals**

En el artículo *ART.2* se muestra el algoritmo *UMEDC* (Unsupervised Method for Event Detection and Classification of signals). Este algoritmo clasifica señales *MC* cuyos eventos poseen



### *Capítulo 3: Objetivos y resultados de cada publicación*

una localización temporal variable. Además, las señales pueden tener diferente duración.

El algoritmo *UMEDC* utiliza *UMED* para la detección de eventos y posteriormente transforma cada señal *MC* en una secuencia temporal.

Las distancias entre todos los pares de secuencias permiten a un algoritmo de Clustering (por ejemplo *HC*) clasificar las secuencias temporales.

Otra forma de proceder consiste en transformar las secuencias en vectores de características con el mismo número de componentes. Por ejemplo, utilizando un *HMM*. De este modo la clasificación de las señales *MC* se reduce a la clasificación del conjunto de vectores de características.

En este artículo también se ha propuesto un nuevo método, con la ayuda del algoritmo *SC*, para transformar un conjunto de secuencias de distintas longitudes en vectores de características con el mismo número de componentes.

El método *UMEDC* se ha evaluado mediante un conjunto de señales electromiográficas (*EMG*) multicanal del antebrazo, en donde se conocen a priori los eventos principales a detectar (los movimientos de la mano). La evaluación externa no requiere el uso de un índice de validación, ya que el número de señales no es elevado y el resultado obtenido se corresponde exactamente con el esperado. Se

### *Capítulo 3: Objetivos y resultados de cada publicación*

puede comprobar fácilmente que cada señal *EMG* con la misma secuencia de movimientos de la mano pertenece al mismo cluster.

Como se ha mencionado, el método *UMED* utiliza el algoritmo jerárquico (*HC*) para el Clustering y el índice *S\_dbw* para determinar de forma óptima los eventos. Ante la gran cantidad de algoritmos de Clustering e índices de validación interna (que también permiten determinar el *NO*), la combinación de *HC* y *S\_dbw* ha proporcionado buenos resultados. El índice *S\_dbw* es uno de los más completos pues analiza los clusters creados teniendo en cuenta su compacidad, separación y densidad. Además de este índice, el algoritmo *HC* presenta un coeficiente de evaluación propio denominado Coeficiente de Correlación Cofenético (*CCC*) (explicado en la Sección 2.2 del *ART.1*). El *CCC* evalúa la calidad del resultado de un algoritmo *HC* sin estar relacionado con la definición de compacidad y separación entre clusters. Este índice se ha utilizado para acelerar el procesado del algoritmo *UMED* y *UMEDC* (véase la sección 3.3).

#### **3.3 Mejoras para *UMED* y *UMEDC***

En las distintas señales a las que se ha aplicado *UMED*, se ha observado que la clasificación de los estados en una señal utilizando una ventana óptima (correspondiente al óptimo *S\_dbw*) no es muy diferente a la obtenida para la ventana en la que *CCC* alcanza un máximo (donde *HC* tiene un mejor comportamiento). Esto es así

### *Capítulo 3: Objetivos y resultados de cada publicación*

porque las amplitudes de las ventanas y/o los valores del índice  $S_{dbw}$  correspondientes no son muy diferentes. La diferencia principal entre ambas clasificaciones reside en que la clasificación con la ventana del óptimo  $S_{dbw}$  puede diferenciar mejor estados que "funcionalmente tienen un origen cercano". Este hecho puede observarse en *ART\_1* (Fig. 6) respecto a la precisión con la que *UMED* diferencia los movimientos de la cara. Como se explica a continuación, esta particularidad se puede aprovechar para que *UMEDC* procese las señales más rápidamente.

*UMEDC* es un método lento cuando tiene que clasificar un gran número de señales. La parte lenta del método *UMEDC* aparece cuando éste utiliza *UMED* en cada señal para detectar los eventos de forma óptima. El tiempo de procesado es importante y varía según la duración de la señal.

Por otro lado, la caracterización global de los estados (etapa 6 de *UMEDC* en *ART.2*) puede fusionar algunos estados. Entre estos se encuentran los anteriormente descritos como "parecidos desde un punto de vista funcional". Dado que esta fusión va a estar presente, una forma alternativa de trabajar consiste en detectar los eventos de cada señal con la ventana en la que el *CCC* es máximo, ahorrando el tiempo en la búsqueda de la ventana con el óptimo  $S_{dbw}$ . Este cambio acelera de forma muy significativa la clasificación de series temporales mediante *UMEDC*. Por otro lado, esto no impide que en

### *Capítulo 3: Objetivos y resultados de cada publicación*

la caracterización global de los estados (con la ventana del máximo *CCC*) puedan aparecer más fusiones.

Por consiguiente, cuando se trata de aplicar *UMED* sobre una señal, o *UMEDC* sobre un conjunto pequeño de series temporales, se selecciona una ventana óptima. Por el contrario, si se utiliza *UMEDC* sobre un conjunto importante de series temporales, es aconsejable utilizar la ventana en la que *CCC* es máximo con el fin de obtener un procesado rápido de la clasificación no supervisada.

Por otro lado, respecto a *UMED* y la selección de la ventana óptima, no se puede descartar para una señal que el mínimo de  $S_{dbw}$  sea único entre todas las ventanas analizadas. En tal caso, el algoritmo elige la ventana con el mínimo  $S_{dbw}$  más próximo a la ventana del máximo valor para *CCC*.

Además del procedimiento explicado en *ART.1* y *ART.2*, también se pueden considerar otras estrategias para encontrar la ventana con el mínimo  $S_{dbw}$ . Por ejemplo, empezar a buscar a partir de la ventana con el máximo *CCC*.

#### **3.4 Análisis de *ART.3*: Determination of the optimal number of clusters using a spectral clustering optimization**

Junto al algoritmo *HC*, también se han estudiado otros algoritmos de Clustering para ser integrados dentro de *UMED*. Uno de los algoritmos de Clustering más completos es el *SC*. Sin embargo, los métodos para determinar el *NO* adaptados al *SC* son escasos, no

### *Capítulo 3: Objetivos y resultados de cada publicación*

funcionan como los índices clásicos de validación interna y no están bien adaptados a cualquier tipo de datos.

El artículo *ART.3* muestra un nuevo método denominado Spectral Global Silhouette (*GS*) para la determinación del *NO* en un conjunto de objetos utilizando el *SC*. El *NO* se calcula directamente a partir de las nuevas representaciones de los objetos que resultan de *SC*.

El método combina el índice de validación interna Silhouette con el concepto de “Local Scaling” (ambos explicados en la Sección 2 del *ART.3*). Se utiliza una versión simplificada del índice Silhouette (*SS*) para reducir el procesado del algoritmo *GS*.

Además de proponer *GS*, en *ART.3* se han descrito tres nuevos métodos de segmentación de imágenes (*GSWA*, *GSWB*, Nyström\_ *WB*) para mostrar cómo aplicar *GS* cuando el número de objetos a agrupar es elevado. En general, *GSWA* obtiene peores resultados que *GSWB* y Nyström\_ *WB*. Estos últimos se diferencian principalmente del primero en la estrategia utilizada (*WA* o *WB*) para obtener el *NO* mediante *GS*. *WA* reduce el número de datos mediante interpolación mientras que *WB* divide los datos en bloques y selecciona de forma proporcional los más significativos.

Una vez que se determina *NO*, los resultados de *GSWB* y Nyström\_ *WB* son parecidos aunque *GSWB* segmenta la imagen más rápidamente.

El método *GS* se ha podido evaluar mediante inspección visual, tanto con datos sintéticos como con métodos de segmentación.

### *Capítulo 3: Objetivos y resultados de cada publicación*

Además, *SS* proporciona un valor para cada segmentación que permite evaluar cuantitativamente las segmentaciones de los distintos métodos para una misma imagen.

El método *SC* junto a *GS* también puede trabajar con grupos de objetos de diferente densidad.

Dado que el *SC* presenta cierta superioridad a otros algoritmos con algunos tipos de datos (por ejemplo datos que presentan clusters no convexos), la combinación *SC + GS* es útil e interesante. Sin embargo en relación a *UMED* se mantuvo la combinación *HC* y *S\_dbw* ya que, además de proporcionar resultados correctos, es posible utilizar el coeficiente *CCC*. Como se ha explicado anteriormente, este coeficiente proporciona una medida de la calidad del resultado obtenido mediante *HC* y esto puede ser utilizado para acelerar los cálculos en *UMED* y *UMEDC*.

#### **3.5 Análisis de *ART.4: An unsupervised method to determine the optimal number of independent components***

Después de aplicar *UMED* sobre una señal, puede interesar suprimir o aplicar algún tipo de filtrado sobre algunos estados según un criterio específico. Por ejemplo, en *ART.2* se suprimieron los estados menos significativos para obtener por cada señal una secuencia temporal adaptada al tipo de clasificación deseada. También en el *ART.1* se menciona la posibilidad de suprimir los segmentos de señal con artefactos, para obtener una señal *EEG* libre

### *Capítulo 3: Objetivos y resultados de cada publicación*

de artefactos. Sin embargo, esta supresión lleva asociada una pérdida de información, ya que el artefacto se mezcla con la señal *EEG*. En este caso es mejor intentar filtrar el artefacto en lugar de optar por la supresión.

Un análisis *ICA* aplicado sobre una porción de señal *MC* (con el mismo tipo de sensores) proporciona un conjunto de componentes independientes. Después de aplicar *UMED* sobre esa señal *MC*, puede ser interesante utilizar *ICA* sobre una porción de señal que incluya algún estado detectado. Por ejemplo, un segmento *EEG* con artefacto.

Mediante el algoritmo *UMED* se detecta el intervalo temporal donde aparece el artefacto. Este hecho facilita la selección de las *CIs* a filtrar ya que las *CIs* relacionadas con el artefacto presentan una actividad significativa en el mismo intervalo detectado. Un experto puede seleccionar aquellas *CIs* no relacionadas con el artefacto para poder reconstruir la señal *EEG* sin el artefacto. Mediante este proceso se filtra el artefacto de la señal *EEG*.

En general, el número de *CIs* que se obtiene en un *ICA* puede no ser el óptimo. Con el fin de filtrar y/o analizar las *CIs* de forma adecuada, es importante trabajar con un número óptimo de componentes independientes (*NOCI*). El *ART.4* presenta un nuevo método *LCC* no supervisado para obtener el número óptimo de componentes independientes (*NOCI*) de forma automática. Utiliza el algoritmo *JADE* para *ICA*.

### *Capítulo 3: Objetivos y resultados de cada publicación*

El algoritmo *LCC* ha sido evaluado utilizando señales sintéticas construidas a partir de distintas mezclas de un conjunto de señales independientes. También se ha comparado con otros métodos.

A diferencia de otros métodos, *LCC* no necesita una representación gráfica para determinar *NOCI*. Tampoco requiere información previa o agrupar los canales en grupos, ni tiene limitaciones respecto al tipo de *CIs* a encontrar.

Sin embargo, *JADE* limita las posibilidades de *LCC*. Dependiendo de la memoria disponible y las capacidades de la *CPU*, *JADE* puede encontrar dificultades cuando el *NOCI* a encontrar es alto. Además, como cualquier otro método, el número de canales de la señal *MC* debe de ser superior a *NOCI*.





## CAPÍTULO 4



---

### ART.1: UNSUPERVISED EVENT CHARACTERIZATION AND DETECTION IN MULTICHANNEL SIGNALS: AN EEG APPLICATION

---

Resumen:

- *Revista:* Sensors
- *Impact Factor (2015):* 2.033; *5-Year Impact Factor:* 2.437
- *Selección de la primera página del artículo publicado:*

sensors

*Article*

### Unsupervised Event Characterization and Detection in Multichannel Signals: An EEG application

Angel Mur <sup>1,\*</sup>, Raquel Dormido <sup>1</sup>, Jesús Vega <sup>2</sup>, Natividad Duro <sup>1</sup> and Sebastian Dormido-Canto <sup>1</sup>

<sup>1</sup> Department of Computer Sciences and Automatic Control, UNED, Juan del Rosal 16, 28040 Madrid, Spain; raquel@dia.uned.es (R.D.); nduro@dia.uned.es (N.D.); sebas@dia.uned.es (S.D.-C.)

<sup>2</sup> National Fusion Laboratory by Magnetic Confinement, CIEMAT, Complutense 40, 28040 Madrid, Spain; jesus.vega@ciemat.es

\* Correspondence: a.r.m.@outlook.fr; Tel: +34-91-398-7192

Academic Editors: Steffen Leonhardt and Daniel Feichmann  
Received: 4 March 2016; Accepted: 21 April 2016; Published: 23 April 2016

**Abstract:** In this paper, we propose a new unsupervised method to automatically characterize and detect events in multichannel signals. This method is used to identify artifacts in electroencephalogram (EEG) recordings of brain activity. The proposed algorithm has been evaluated and compared with a supervised method. To this end an example of the performance of the algorithm to detect artifacts is shown. The results show that although both methods obtain similar classification, the proposed method allows detecting events without training data and can also be applied in signals whose events are unknown *a priori*. Furthermore, the proposed method provides an optimal window whereby an optimal detection and characterization of events is found. The detection of events can be applied in real-time.

**Keywords:** artifacts; EEG; event characterization; event detection; unsupervised classification

---

Article

## Unsupervised Event Characterization and Detection in Multichannel Signals: An EEG application

Angel Mur <sup>1,\*</sup>, Raquel Dormido <sup>1</sup>, Jesús Vega <sup>2</sup>, Natividad Duro <sup>1</sup> and Sebastian Dormido-Canto <sup>1</sup>

<sup>1</sup> Department of Computer Sciences and Automatic Control, UNED, Juan del Rosal 16, 28040 Madrid, Spain; raquel@dia.uned.es (R.D.); nduro@dia.uned.es (N.D.); sebas@dia.uned.es (S.D.-C.)

<sup>2</sup> National Fusion Laboratory by Magnetic Confinement, CIEMAT, Complutense 40, 28040 Madrid, Spain; jesus.vega@ciemat.es

\* Correspondence: a.r.m@outlook.fr; Tel.: +34-91-398-7192

Academic Editor: Steffen Leonhardt and Daniel Teichmann

Received: 4 March 2016; Accepted: 21 April 2016; Published: date

**Abstract:** In this paper, we propose a new unsupervised method to automatically characterize and detect events in multichannel signals. This method is used to identify artifacts in electroencephalogram (EEG) recordings of brain activity. The proposed algorithm has been evaluated and compared with a supervised method. To this end an example of the performance of the algorithm to detect artifacts is shown. The results show that although both methods obtain similar classification, the proposed method allows detecting events without training data and can also be applied in signals whose events are unknown *a priori*. Furthermore, the proposed method provides an optimal window whereby an optimal detection and characterization of events is found. The detection of events can be applied in real-time.

**Keywords:** artifacts; EEG; event characterization; event detection; unsupervised classification

---

### 1. Introduction

Electroencephalography (EEG) is a non-invasive method to record electrical activity of the brain using sensors distributed along the scalp. Signals detected by an EEG with no connection to a specific brain activity are called artifacts. These are usually classified as physiological and non-physiological artifacts. Physiological artifacts are generated by the patient and non-physiological artifacts can arise from outside of the body (*i.e.*, equipment, environment, *etc.*).

In many cases the information that is hidden behind the physiological artifacts is relevant to a proper diagnosis. Think, for

instance, about early detection of mental fatigue or in monitoring stress levels. Others health applications need EEG signals without artifacts contamination in order to reduce the misinterpretation of the EEG and limit the potential for adverse clinical consequences. In this case brain computer interfaces need to filter artifacts in real time [1,2]. Consequently, it is necessary, especially for long recordings, to develop methods to detect and correctly identify artifacts either for analyzing or removing them. This is the main goal of this paper.

There are some approaches for detecting and removing artifacts using EEG recordings. Statistical methods as SCADS (Statistical Control of Artifacts in Dense Array Studies) [3] and FASTER (Fully Automated Statistical Thresholding for EEG Artifact Rejection) [4] detect and remove artifacts to analyze event-related potentials. The packages EEGLAB [5] and FieldTrip [6] include some routines to detect EEG artifacts but some threshold values need to be defined. Reference [7] is a practical example of how threshold values are used for detection of EEG artifacts in polysomnographic recordings.

DETECT [8] is a toolbox for detecting and identifying artifacts based on the supervised classifier support vector machine (SVM). Note that other supervised methods do not carry out an identification process [9–12]. An advantage of DETECT over other methods is that the user is not required to manually define threshold values (with the exception of a certainty value applied to the classification outcome to reduce false positives). Moreover it captures the frequency and duration of artifacts. DETECT only finds artifacts included in the training process and the quality of this detection depends on the quality of the training data.

In [13], the authors present a practical example of how a SVM classifier is used to detect artifacts arising from head-movements. Reference [14] describes a novel application of one-class support vector machine novelty detection for detecting seizures in humans.

In this paper we first propose a new, simple, and effective unsupervised method (UMED) to characterize and detect events in multichannel signals. Then the proposed method is applied to detect artifacts in EEG recordings. We use the term “artifact detection” interchangeably with the term “event detection” although artifact is more closely associated with the EEG recordings.

Some advantages of the proposed method are the following: (i) it does not need to manually specify any threshold value; (ii) it allows the measurement of the frequency and duration of the events; (iii) as an unsupervised method it does not need training data; (iv) it can be applied to signals with *a priori* unknown events; (v) it provides an optimal detection technique to characterize the events by finding an optimal window; (vi) it can be used in real-time; and (vii) it provides the number of independent components (NIC) required to remove artifacts by means of an independent component analysis (ICA) [2,15,16].

This paper is organized as follows. Section 2 presents some terms and theoretical background used to implement UMED. The proposed unsupervised method to characterize and detect events in multichannel signals is presented in Section 3. In Section 4, we test this method using an EEG recording with artifacts. Finally in Sections 5 and 6, a discussion and conclusions of the paper are respectively presented.

## 2. Background

### 2.1. Events and Multi-Channel Signals

In this section, we formally define the terms “event” and “multichannel signal”. Both are linked to the concept of temporal sequence. Temporal sequence refers to a sequence of happenings or events in a time interval. An event is something that happens at a time  $t_E$  and it reflects some kind of change in a temporal evolution. An important characteristic of temporal sequences is the order in which events take place. Another important characteristic is the duration  $d_E$  between  $t_E$  and the next event.

A temporal sequence is normally represented by a series of nominal symbols from a particular alphabet. Every event is characterized by a symbol. In this way, an event with symbol  $E$  is described by two elements  $(E, t_E)$ . The three elements  $(E, t_E, d_E)$  represent a state. A temporal sequence  $P$  made of  $u$  events  $E^i$  for  $i = 1, \dots, u$  can be described by a  $u$ -tuple of states  $P = \langle (E^1, t_E^1, d_E^1), (E^2, t_E^2, d_E^2), \dots, (E^u, t_E^u, d_E^u) \rangle$ .

In general, the number and temporal locations of events can be unknown within a temporal sequence. This means that events have to be recognized and located inside the sequence. It occurs, for example, in the monitoring of a physical system with  $Q$  sensors for a period of time  $[T_1,$

$T_2$ ]. In this case, every sensor represents a channel and simultaneously provides a signal describing part of the whole information. Signals of this type that are generated by multiple sensors are called multichannel (MC) signals. In other words, a MC signal refers to a set of signals that show cross-channel similarity or correlation. A MC signal with  $Q$  channels is represented by a vector  $X(t) = [CH_1(t) CH_2(t), \dots, CH_Q(t)]$  where  $CH_q(t)$  is the signal of the channel  $q$  for  $q = 1, 2, \dots, Q$ . A MC signal  $X(t)$  with events can also be described, in compact form, as a temporal sequence  $P$ .

## 2.2. The Hierarchical Clustering, the Cophenetic Correlation Coefficient, and the $S\_Dbw$ Validity Index

In this section we briefly describe three concepts used in the implementation of the proposed algorithm: the hierarchical clustering (HC) algorithm [17], the Cophenetic Correlation Coefficient (CCC) [18], and the  $S\_Dbw$  Validity Index [19].

The HC groups data over a variety of scales by creating a cluster tree or dendrogram. It follows this general procedure: (1) find the similarity or dissimilarity between every pair of objects in the data set; (2) group the objects into a binary, hierarchical cluster tree (linkage); (3) determine where to cut the hierarchical tree into clusters.

In a dendrogram, the cophenetic distance between two objects is represented by the height of the link at which those two objects are first joined. That height is the distance between the two subclusters that are merged by that link. The CCC for a cluster tree is defined as the linear correlation coefficient between the cophenetic distances obtained from the tree, and the similarities (or dissimilarities) used to construct the tree. The CCC is a measure of how faithfully a dendrogram maintains the original pairwise distances. The magnitude of CCC should be very close to 1 to achieve a high-quality solution of the HC algorithm.

Given a dataset, if  $x(O_i, O_j)$  is the distance between the  $i$ th and  $j$ th object,  $t(O_i, O_j)$  is their cophenetic distance,  $\bar{x}$  is the average of the  $x(O_i, O_j)$  and  $\bar{t}$  is the average of the  $t(O_i, O_j)$ , then the CCC is given by Equation (1):

$$CCC = \frac{\sum_{i < j} (x(O_i, O_j) - \bar{x})(t(O_i, O_j) - \bar{t})}{([\sum_{i < j} (x(O_i, O_j) - \bar{x})^2][\sum_{i < j} (t(O_i, O_j) - \bar{t})^2])^{1/2}} \quad (1)$$

The  $S\_Dbw$  validity index is used for measuring “goodness” of a clustering result. Its definition is based on cluster compactness and separation but it also takes into consideration the density of the clusters. Lower index value indicates better clustering schema.

Given a set of  $k$  clusters  $G_i$  ( $i = 1, \dots, k$ ) of the dataset  $DS$ , the  $S\_Dbw$  index is defined in the following way:

$$S\_Dbw(k) = Scat(k) + Dens\_bw(k) \quad (2)$$

where  $Scat$  is the intra-cluster variance that measures the average scattering of clusters and it is described by:

$$Scat(k) = \frac{1}{k} \sum_{i=1}^k \frac{\|\sigma(G_i)\|}{\|\sigma(DS)\|}$$

$\sigma(DS)$  is the variance of the dataset  $DS$  and  $\sigma(G_i)$  is the variance of the cluster  $G_i$ .

The inter-cluster density  $Dens\_bw$  is defined as follows:

$$Dens\_bw(k) = \frac{1}{k(k-1)} \sum_{i=1}^k \left( \sum_{j=1, j \neq i}^k \frac{density(G_i \cup G_j)}{\max[density(G_i), density(G_j)]} \right)$$

where  $density(G) = \sum_{i=1}^{|G|} f(x_i, \mu)$ ,  $\mu$  is the center of the cluster  $G$ ,  $|G|$  is the number of objects in  $G$ , and the function  $f(x_i, \mu)$  is defined by:

$$f(x, \mu) = \begin{cases} 0 & \text{if } distance(x, \mu) > \frac{1}{k} \left( \sum_{i=1}^k \|\sigma(G_i)\| \right)^{1/2} \\ 1 & \text{otherwise} \end{cases}$$

If  $G = G_i \cup G_j$ , then  $\mu$  is the middle point of the line segment that is defined by the  $\mu_i$  and  $\mu_j$  clusters centers.

### 3. Method for the Characterization and Detection of Events in Multichannel Signals

In this section, we propose an unsupervised method (UMED) to characterize and detect the events of a MC signal of duration  $[T_1, T_2]$  whose events are a priori unknown. A small temporal window slides along the interval  $[T_1, T_2]$ . For each window displacement, it is obtained a feature vector of  $S$  variables which picks up the behavior of the MC signal. A feature vector unsupervised classification is carried out, which allows automatic characterization and detection of the events. The quality of this characterization (and the number of events detected) depends significantly on the size of the sliding window. This is why the algorithm searches for an optimal window size that provides the maximum compactness and separation of the content between the events.

The UMED detects and characterizes events by means of an optimal unsupervised classification. The  $S\_dbw$  index and the HC algorithm are used to find the best window along with its unsupervised classification characterized by its optimal number of clusters.

Given a MC signal  $X$  for a period of time  $[T_1, T_2]$  with unknown events to be characterized and detected, the steps of UMED can be summarized as follows:

**Step 1: Specification of windows.** The algorithm uses a sliding window approach. Figure 1 shows the details of the sliding window in a MC signal  $X$  along the interval  $[T_1, T_2]$ . If the window  $W_i$  of size  $L_w$  is defined by the interval  $[t_1, t_2]$  then the window  $W_{i+1}$  is defined by  $[t_1 + d, t_2 + d]$  with  $d > 0$ . As first step, a set of  $IN$  windows is selected for a specific  $L_w$  and  $d$ .

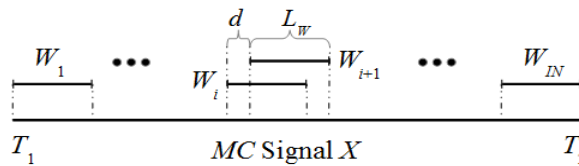


Figure 1. Different windows along the MC signal X.

**Step 2: Feature matrix (FV) calculation.** In this step a matrix  $FV$  of  $IN$  rows is generated. Each row  $i$  of  $FV$  represents a  $Q$ -feature vector of the



window  $W_i$ . So first we calculate for each channel  $q$  and window  $W_i$  a feature vector of size  $s$ . Then, these features vectors are concatenated to form a unique  $Q$ -feature vector  $Fv_i$  (with  $S = Q \times s$  features) for each  $W_i$ . At last, all the feature vectors  $Fv_i$  (for  $i = 1, \dots, IN$ ) are saved in a matrix  $FV$  of size  $IN \times S$  concatenating them vertically.

Two autoregressive coefficients are good candidates to form the feature vector for each channel  $q$  and window  $W$ . The coefficients  $a_1$  and  $a_2$  of an autoregressive model  $AR(2)$  of a window  $W(r)$  for a single channel can be written as:

$$W(r) = \sum_{r=1}^2 a_r W(r-i) + e(r) \quad (3)$$

where  $e(r)$  is zero-mean white noise. The  $a_i$  coefficients can be estimated using Burg's method [20]. Normally the  $Q$ -feature vector with a  $AR(2)$  (of length  $Q \times 2$ ) offers superior or equal performance to detect artifacts in EEG recordings than a  $AR(p)$  with  $p > 2$  [8,21], and it is computationally less intensive.

**Step 3: Selection of a window size  $L_w$  using the CCC.** In this step, first the CCC of the  $IN$  rows of the matrix  $FV$  is calculated. If CCC reaches approximately a particular threshold  $U$ , it is selected the value of  $L_w$ . This value is named  $L_w^0$  and its feature matrix  $FV^0$ . In case CCC is much lower than  $U$ , we go back to the step 1 to choose a larger  $L_w$  and then a new  $FV$  and CCC are tested.

The CCC threshold  $U$  is defined as 0.85. This value comes from experience and guarantees that the HC algorithm will find a significant classification. From a practical point of view, this threshold is used to reduce the grid search for the window length in step 4. For small window sizes (some dozens of samples) the CCC value can be  $< 0.85$ . In this case the CCC is used to determine a minimum window size from which the step 4 will start. In step 4, the  $S\_dbw$  takes the responsibility to calculate an optimum window size. A low value of  $S\_dbw$  comes with a high CCC value. However, the CCC is not defined by compactness and separation of groups and it cannot be used to determine an optimal group number.

For each window size the processing time of CCC is low. So a possible strategy to determine  $L_w^0$  is to select a small group of window

sizes and check if the  $CCC > 0.85$  is respected from a particular window size.

**Step 4: Selection of an optimal window size ( $L_w^H$ ) and its unsupervised classification.** In this step, first the size of the window is increased  $N_L$  times by means of  $L_w^m = L_w^0 + m \times D_L$  for  $m = 0, \dots, N_L$  and the integer  $D_L \geq 1$ . Second, for each  $L_w^m$ , it is calculated its feature matrix  $FV^m$ . Third, the HC algorithm is applied  $N_G$  times to classify the rows of each matrix  $FV^m$  in different partitions *i.e.*, in 2, 3, ...,  $N_G + 1$  groups. Fourth, for each partition, its  $S\_Dbw_g^m$  index (with  $g = 2, \dots, N_G + 1$ ) is calculated. The minimum value of the index, named  $Min^m$ , determines the optimal number of groups  $No^m$ . The minimum value  $Min^H$  between the *Min*'s determines the optimal window size  $L_w^H$ , the optimal feature matrix  $FV^H$  and the optimal number of groups  $No^H$ .

The selected  $N_L$  value allows to check a significant number of window sizes. The MC signal  $X$  has an initial period of time (IPT) where there are no events. If IPT is known, then  $Lw^{NL}$  has to be  $< IPT$ . If IPT is not known, then it is possible to add at the start of  $X$  a synthetic portion of signal without events. When the size  $L_w^H$  is near to  $Lw^{NL}$  a bigger value of  $N_L$  is chosen (with the help of a synthetic IPT, if necessary) to ensure that  $L_w^H$  is well surrounded by other suboptimal sizes.

The selected  $N_G$  value allows to check a significant number of partitions. In case the optimal number  $No^m$  is near to  $N_G$ , a bigger value of  $N_G$  is chosen to ensure that  $No^m$  is well surrounded by other suboptimal group numbers.

**Step 5: Event Detection by means of the optimal unsupervised classification.** It is accomplished the unsupervised classification for the predetermined number of groups  $No^H$ . The condition  $CCC > U$  and the selection of  $Min^H$  allow the HC algorithm to find a high-quality classification with the best group compactness and separation taking into consideration the density of the clusters.

Once all the windows of size  $L_w^H$  have been classified into  $No^H$  clusters, the different events can be detected along the MC signal  $X$ . After this classification, we will use, preferably, the term interval instead of the window to stress the window time interval.

The events are characterized by the group numbers of the optimal classification. These are placed at the beginning and the end of each group of consecutive intervals  $[W_i, W_{i+1}, \dots, W_j]$  where  $(1 \leq i \leq j \leq IN)$  with the

same group number. The beginning and the end can be defined using the next formula [8]:

$$\left[ (i-1) \times d + M \times L_w - \frac{d}{2}, (j-1) \times d + M \times L_w + \frac{d}{2} \right] \quad (4)$$

where  $i$  is the number of the first intervals  $W_i$ ,  $j$  is the number of the last interval  $W_j$ ,  $d$  is the slide width, and  $M = 6/7$ . The product  $L_w \times 6/7$  is more accurate than using the midpoint with  $M = 1/2$ . Equation (4) uses only  $d$  samples per interval along the signal without overlapping. In this way, intervals between events are uniquely labeled regardless of whether events are placed in the intersection of two overlapping windows classified in a different way. Equation (4) is used to plot the events along the EEG signal. It finds the events' temporal locations along the EEG signal and, thus, it is possible to study their behavior.

#### 4. Testing the Algorithm

##### 4.1. EEG multichannel Signal with Artifacts

The proposed algorithm has been tested to detect artifacts in multichannel EEG signals. The test signal is an EEG recording of 8 min duration used in [8] to highlight the utility of the DETECT toolbox. The data are sampled at 256 Hz using a 64-channel Biosemi Active Two System (Amsterdam, Netherlands). Figure 2 shows the channel  $CH_1(t)$  of the EEG recording. Our test uses all of the channels.

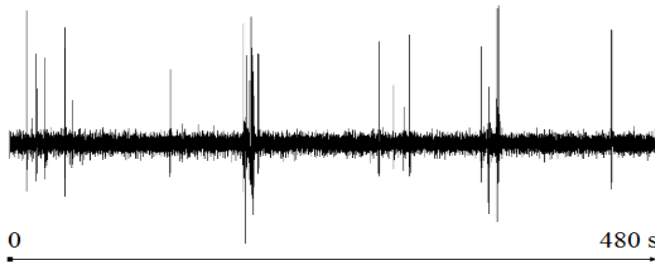


Figure 2. The channel  $CH_1(t)$  of the 64-channel EEG.

The artifacts are categorized into six classes [8]: None (NN), Jaw Clench (JC), Jaw Movement (JM), Eye Blink (EB), Eye Left Movement (ELM), and Eye Up Movement (EUM). In reference [8] a SVM approach is used for detection of artifacts. This supervised method (SM) is trained with a balanced set of 20 trials for each type of artifact along with a window of  $L_w = 128$  samples and a slide width of  $d = 32$  samples. Each window of a channel is characterized by the first two autoregressive coefficients [21]. A number of 3837 windows (or intervals) are analyzed and classified. Furthermore, in [8] a certainty thresholding policy is applied to this classification to remove false positives in the data. The Table 1 summarizes the SM outcome after using a value of 0.5 to threshold the certainty.

**Table 1.** Number of intervals per group using the SM. The groups are None (NN), Jaw Clench (JC), Jaw Movement (JM), Eye Blink (EB), Eye Left Movement (ELM), and Eye Up Movement (EUM).

Group	Number of Intervals
NN	3715
ELM	12
EUM	21
EB	81
JM	5
JC	3

#### 4.2. Detection and Characterization of Events Using EEG Recordings

In this section, we apply the proposed UMED method presented in Section 3 to detect and characterize the artifacts in the EEG recording described above. A comparison with the results summarized in Table 1 is also given. Both approaches can be compared because they use the same AR feature vectors and a similar sliding window approach.

The first two autoregressive coefficients have been used as feature vector per window and channel. For  $L_w = 128$  samples, the CCC is 0.8510 and  $N_0 = 7$ . The Table 2 shows (for the step 4) the CCC values, the  $Min$  values and the optimal number of clusters for different  $L_w > 128$  samples starting at  $L_w = 135$  and  $D_L = 5$ .

**Table 2.** Characteristic parameters for the selection of the optimal window (step 4): The window size  $L_w$ , the optimal number of clusters  $No$ , the Cophenetic Correlation Coefficient CCC, and the minimum value of the  $S\_Dbw$  index  $Min$ .

$L_w$	135	140	145	150	155	160	165
CCC	0.8413	0.8564	0.8473	0.8475	0.8658	0.8616	0.8679
$No$	7	7	7	7	7	7	6
$Min$	0.97	0.80	0.85	0.90	0.77	0.82	1.05

From the data presented in Table 2 it is easy to find the minimum value of the index ( $Min$ ) among the minimums,  $Min^H = 0.77$ ; the optimal number of clusters  $No^H = 7$  for  $L_w^H = 155$  samples, and  $CCC^H = 0.8658$ . Then, an unsupervised classification (HC algorithm), as stated in step 5, allows finding of the optimal content of the seven groups.

We use a principal component analysis (PCA) [22] to display the clusters found on a 2D representation. PCA is a projection method that re-expresses a collection of correlated variables into a smaller number of variables called principal components, which maximizes the variance of the data.

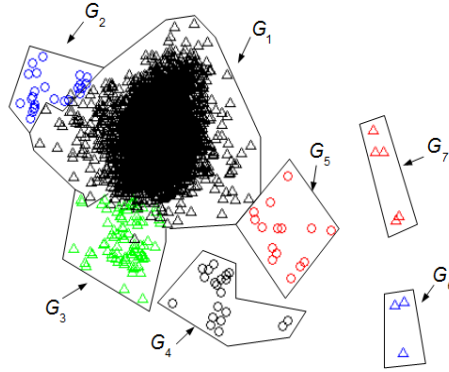
If the original data formed by  $n_v$  vectors of  $l$  features are stored in a matrix  $O_D(l \times n_v)$ , then the PCA transform is:

$$T_D = A(O_D - m_{OD}) \tag{5}$$

where  $m_{OD}$  is the mean of the original data and  $A$  is a matrix whose rows are the eigenvectors of the covariance matrix  $C_{OD} = (1/n_v)O_D O_D^T$ .

Our PCA is carried out using the Equation (5) and  $O_D = (FV^H)^T$ . Then, it is selected the submatrix  $T_D^2$  made up of the first two principal components (the first two rows of  $T_D$ ). The column  $k$  of  $T_D^2$  provides a new feature vector of only two elements to the window  $W_k$ .

Figure 3 shows the seven optimal clusters found. The interval  $z$  is represented by a point whose coordinates are  $T_D^2(1, z)$  and  $T_D^2(2, z)$ . Each cluster is distinguished by a specific shape and color. Table 3 summarizes the UMED outcome for  $L_w^H = 155$ .

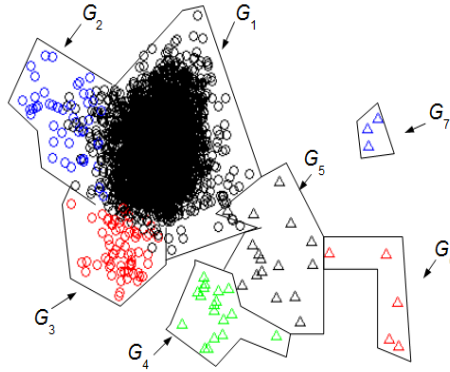


**Figure 3.** Groups of intervals found using the unsupervised classification (UMED) and a PCA. The  $L_w^H$  has 155 samples. The first two principal components contain 55% of the full information. The different clusters are characterized but not identified.

**Table 3.** Number of intervals per group using the UMED with  $L_w^H = 155$  samples.

Group	Number of Intervals
$G_1$	3691
$G_2$	24
$G_3$	77
$G_4$	20
$G_5$	15
$G_6$	3
$G_7$	5

Figure 4 shows the seven clusters found using our UMED for  $L_w = 128$ . Table 4 summarizes the UMED outcome. This result is important to compare UMED with SM.

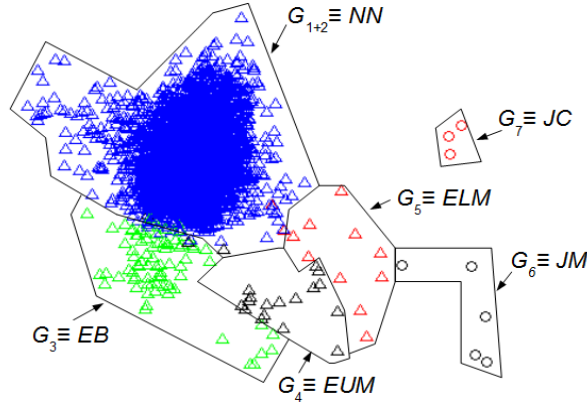


**Figure 4.** Groups of intervals found using the unsupervised classification (UMED) and a PCA. The  $L_w$  has 128 samples. The first two principal components contain 53% of the full information. The different clusters are characterized but not identified.

**Table 4.** Number of intervals per group using the UMED with  $L_w = 128$  samples.

Group	Number of Intervals
$G_1$	3680
$G_2$	46
$G_3$	67
$G_4$	18
$G_5$	16
$G_6$	5
$G_7$	3

Figure 5 shows the 6 clusters found with DETECT [8] using a supervised classification (SM) for  $L_w = 128$  and a value of 0.5 to threshold the certainty. The principal components are the same as those of Figure 4. Table 1 summarizes the SM outcome where the NN intervals are in  $G_{1+2}$ , the EB in  $G_3$ , the EUM in  $G_4$ , the ELM in  $G_5$ , the JM in  $G_6$ , and the JC in  $G_7$ .



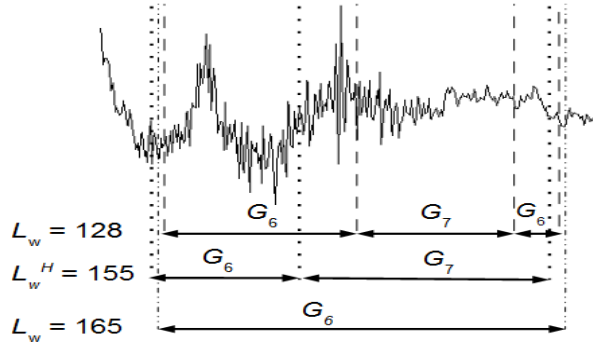
**Figure 5.** Groups of intervals found using the supervised classification (SM) [8] and the first two principal components of the Figure 4. The  $L_w$  has 128 samples. The first two principal components contain 53% of the full information. The different clusters are identified. Using the Table 1: NN intervals are in  $G_{1+2}$ , the EB in  $G_3$ , the EUM in  $G_4$ , the ELM in  $G_5$ , the JM in  $G_6$ , and the JC in  $G_7$ .

Figures 4 and 5 allow the comparison of the UMED and the SM. The UMED finds a group  $G_2$  that is not detected when using SM because the intervals of this group do not form part of the required training process in the SM. Consequently, these intervals are integrated into the  $G_{1+2}$  group that contain the NN intervals. It is observed that SM obtains groups more blurred than the UMED but, in general, the detection of intervals is similar. The only exception are around seven intervals between group  $G_3$  and  $G_4$ , and around six intervals between groups  $G_4$  and  $G_5$ . Moreover, the Normalized Mutual Information (NMI) index [23] allows to compare the intervals' group numbers for both methods. If, for the classification of UMED, the  $G_2$  is included into the  $G_1$ , the NMI between the UMED and SM result is 0.8317. This score is highly consistent since UMED and SM find a similar classification of intervals. A value of NMI close to 1 means that the results of both methods are similar.

The groups  $G_6$  and  $G_7$  have events that are consecutive in time. Figure 6 shows the effect on the classification of these groups using a  $L_w^H = 155$ ,



$L_w = 128$  and  $L_w = 165$  (see also Table 2). By using Equation (4) the events can be temporally located along the EEG signal and, thus, it is possible to study their behavior.



**Figure 6.** Events detected between the samples 46,389 and 46,653 using a  $L_w^H = 155$ ,  $L_w = 128$  and  $L_w = 165$ . The signal is a portion of an EEG channel.

The optimal  $L_w^H$  provides two groups  $G_6$  (intervals 1447, 1448, 1449) and  $G_7$  (intervals 1450, 1451, 1452, 1453, 1454) well separated and these groups are mathematically more significant than the  $L_w = 128$  ones (intervals 1448, 1449, 1450, 1451, 1455 for  $G_6$  and 1452, 1453, 1454 for  $G_7$ ). The  $G_6$  of  $L_w^H$  has one artifact and the one of  $L_w = 128$  two artifacts. The  $L_w = 165$  yields one group  $G_6$  and this is why, in Table 2, its  $No$  is 6.

Figure 6 shows, graphically, how the quality of the results depends on the selected sliding window size.

## 5. Discussion

Artifacts can be either analyzed or filtered from an EEG recording. The UMED, as the SM approach, has been mainly developed to analyze artifacts, to study for instance the frequency and the duration of artifacts. The UMED also provides, in a unsupervised way, the number of different artifacts. For instance, in Figure 6 and for  $L_w^H$ , the number of artifacts is 2: JM and JC. The durations of JM and JC are 0.37 s and 0.62 s, respectively. Both appear once during the entire EEG recording.

In Section 4 the proposed UMED has been evaluated by comparing its classification result with the one obtained with the SM using the same

$L_w$ ,  $d$  and EEG recording. It has been used with the PCA and the NMI to contrast, visually and quantitatively, both classifications. This evaluation is significant because, in addition to the fact that two different methods UMED and SM obtain similar results, the SM method was validated comparing “user labelings” with “automated labelings”. The SM method was validated by measuring the agreement (in seconds) between expert and SM labeling in some EEG recordings. This agreement is, in the worst case, over 80%. The best case reaches about 98%. In this way, we have checked that UMED with the same  $L_w$ ,  $d$ , and EEG recording provides a similar result to the one given by an expert. Therefore, UMED could also be considered as an unsupervised expert system.

Unlike SM, UMED determines an optimal window size  $L_w^H$  which allows an optimal classification based on the  $S\_Dbw$  index. With this index and the HC algorithm, UMED characterizes clusters that have different density and can provide new clusters that SM could not have considered during the training process. The HC computes the distance between two data points and the distance between two clusters (linkage) using, respectively, the Euclidean and the average distance (the average distance between each element in one cluster to every element in the other cluster).

The quality of the SM outcome depends on the quality of the training process but, as UMED, it also depends on the size of the sliding window selected. As said in [8]: “If the window size is too small, then the window may not contain sufficient information to model the feature. On the other hand, if the window is too large, SM will not be able to distinguish closely-spaced events, and temporal precision regarding the exact onset of an event is diminished”. This is also true for UMED. Unlike UMED, SM has no mathematical criteria to decide the sliding window size, except to check different sizes and compare the results with the opinion of an expert. This process seems slower and more tedious than the one of UMED.

We can compare the Figures 3 and 4 with Figure 5 to identify the clusters obtained in Figures 3 and 4. In this case, it is not necessary for an expert to identify the clusters found using UMED. In Figure 4, by means of UMED, the groups  $G_3$  and  $G_4$  are clearly distinguished. On the contrary, in Figure 5, SM does not differentiate well these groups because some Eye Up Movement (EUM) are considered as Eye Blink (EB). Some EUM do not

follow the PCA representation (the lowest green triangles are in  $G_3$  instead of  $G_4$ ). Possibly, a better quality of the trials (used for the training process) would have improved the quality of the SM result. In Figure 3, due to the use of  $L_w^H$ , the compactness and separation of the clusters are better than in Figures 4 and 5. Likewise, in Figure 6, the intervals of Jaw Movement  $G_6$  and Jaw Clench  $G_7$  for  $L_w^H$  are better defined.

The SM can be used in real time to detect artifacts in an EEG recording. The classifier has been trained with a balanced set of trials for each type of artifact. Both, the number of artifacts and their trials have to be selected by an expert. Their size  $L_w$  is the same for all. The SM classifier uses a sliding window with the same size  $L_w$ . In real time, each new interval is classified during the acquisition of the next interval.

The UMED can also work in real time primarily in two ways. The first one (UMED<sub>1</sub>) acts similarly to SM but now the training process of the SVM is different. The UMED automatically detects the number of artifacts and their trials from a significant EEG recording using an optimal window size  $L_w^H$ . Then, an expert only needs to identify one portion of consecutive intervals of each cluster and select a significant number of intervals per cluster. This number has to be adequate to get a balanced set of trials. Once the SVM has been trained with these trials (the result of UMED), it is ready to work in real-time. Similarly to SM, each new interval (from other EEG recording) is classified during the acquisition of the next interval.

The second way (UMED<sub>2</sub>) consists on using UMED as a supervised method without using a SVM. In this case the first step is to get a reference classification from a significant EEG recording similarly to UMED<sub>1</sub>. Unlike UMED<sub>1</sub> that selects a set of intervals for a training process, UMED<sub>2</sub> characterizes a dataset  $DS_2$  using the feature vectors of all the intervals (of size  $L_w^H$ ). The UMED<sub>2</sub> selects  $DS_2$  along with its optimal number of clusters  $No^H$  and its  $S\_Dbw$  index named  $IndN_o$ . An expert only needs to identify one portion of consecutive intervals of each cluster.

Then, UMED<sub>2</sub> is ready to classify new sets of  $N_{int}$  intervals from other EEG recording. Each set of intervals is added to  $DS_2$ . The optimal number of clusters is calculated using the  $S\_Dbw$  index and then it is compared the  $S\_Dbw$  index for  $No^H$  (named  $IndN_o$ ) and the one for  $No^H + 1$  (named  $IndN_1$ ). If the value  $IndN_o < IndN_1$  the solution is the unsupervised classification using HC for  $No^H$  groups. If  $IndN_o > IndN_1$  it is compared

$IndN_1$  with  $IndN_2$  and so on until  $IndN_i < IndN_{i+1}$ . In general, the solution is the classification using HC for  $No_i^H$  groups when  $IndN_i < IndN_{i+1}$ .

Usually with a significant  $DS_2$  (containing all the types of artifacts) each of the  $Nint$  intervals belongs to one of the  $No^H$  groups and  $IndN_o < IndN_1$ . The  $Nint$  intervals are classified at the same time and, after that, their feature vectors are removed from the reference dataset. However, if any of the  $Nint$  intervals are classified in a new cluster, then it can remain in the dataset. The new cluster will be a new type of artifact, but it is not identified. This is similar to a SVM that only identifies artifacts included in the training process. In real-time, every  $Nint$  intervals are classified during the acquisition of the next  $Nint$  intervals.

The SM method identifies the artifacts. The UMED method characterizes and detects the artifacts. The identification process is also possible by means of  $UMED_1$  and  $UMED_2$ . Thanks to UMED the training process in  $UMED_1$  and  $UMED_2$  is simpler and faster than the one of SM. It is understood that  $UMED_1$  and  $UMED_2$  can also be used for offline applications.

The UMED does not reduce the feature vectors dimensionality by using for example PCA. The PCA is only used to visually contrast the SM and UMED outcomes. In addition to the loss of information, the use of PCA would be a disadvantage for  $UMED_1$  and  $UMED_2$ .

Some real-time EEG applications use a small number of electrodes. The use of UMED is not limited by the number of channels. With a lower number, the computational effort decreases, though the quality of the result can change. Usually, with fewer channels, discrimination between artifacts decreases so different artifacts could be classified into the same group. If the goal is the correct identification of the artifacts, then, this rule applies: the more channels, the better the discrimination of artifacts.

Artifacts are superimposed on the real EEG signal. Once they have been detected, they can be directly removed by cutting the segments of the signal where there are overlaps. In this case there is a loss of information because those segments include a part of the real EEG signal. The UMED (that detects artifacts and the portions of signals without artifacts (PWA)), can also be used along with other algorithms, such as ICA [15,16], specialized to filter artifacts leaving the full EEG signal with useful information. The PWA can lose information after using ICA. By replacing the PWA after using ICA with the original PWA detected using

UMED ensures the maximum of useful information along with the EEG signal without artifacts. UMED also provides the *NIC* used in an ICA that corresponds to the number of clusters. For the full EEG signal of Figure 1,  $NIC = 7$ . For a specific EEG segment, this number will depend on the artifacts detected by UMED in that portion of signal. In Figure 6,  $NIC = 3$ .

There is a large number of recent research papers that present artifact de-noising of EEG signals. Reference [24] presents an algorithm for removing peak and spike noise from EEG. This is based on filtering and thresholding the analytic signal envelope. Reference [25] presents an unsupervised algorithm that uses modified multiscale sample entropy and Kurtosis to automatically identify the independent eye blink artefactual components and, subsequently, denoise these components using biorthogonal wavelet decomposition. This method neither requires manual identification for artefactual components nor an additional electrooculographic channel. The method *FORCe* [26] is an artifact removal method developed for use in brain-computer interfacing. It is based upon a combination of wavelet decomposition, independent component analysis, and thresholding. The method LAMIC [27] does not outperform *FORCe* but it is interesting because the artifact removal from the EEG has been performed by means of a clustering algorithm. The EEMD-ICA approach [28] removes artifacts preserving more information than other classical ICA methods.

The UMED has been used to study physiological artifacts without the presence of electrical interferences. Modern acquisition systems allow for reducing that kind of interference. However, for a specific application, it could be interesting to check the UMED behavior in the presence of non-physiological artifacts. Usually, the UMED will create new clusters if the electrical artifacts are strong and located in specific portions of the signal. In case the electrical interference is weak along the full EEG signal, there would be no new clusters related to that non-physiological artifact and the UMED performance would not change. When the electrical interferences are not the subject of study, the best choice is to avoid them or filter them during the acquisition process. That is imperative when the interference is strong and distributed continuously along the full EEG signal

The UMED method has been presented using an EEG recording. The artifacts are very well-differentiated from the cognitive EEG part without

artifacts. This cognitive part is only related to the thinking and its high variability is concentrated in a unique cluster that is the biggest. This cluster is not affected using different subjects since the *AR* coefficients are scale- and location-invariant [8,21]. Normally, there is also a unique cluster per artifact from multiple subjects. It may happen that a type of artifact, from a subject with very different morphology, could be grouped in a different cluster. In this case, we could find more than a cluster per artifact, although the distinction between the different artifacts remains. This possibility can diminish as the number of subjects increases.

The *AR*(2) coefficients are good candidates as feature vectors because they allow differentiation between artifacts in the EEG signal and are also scale- and location-invariant. Furthermore, with *AR*(2), the cognitive part without artifacts is well differentiated and is grouped in a single cluster.

The *AR* coefficients are useful to detect artifacts but they cannot detect cognitive events. In theory, once an EEG recording has been cleaned of artifacts, the UMED could be used along with other type of feature vectors that contain spatio-temporal information to detect cognitive events. The UMED outcome for these feature vectors could be evaluated by using a non-invasive optical technique as Near-infrared spectroscopy (NIRS) [29–32].

## 6. Conclusions

In this paper a new unsupervised method of classification of events in multichannel signals from a sensor system has been presented (UMED). This classification allows characterization and detection of the events. In particular, UMED has been applied to detect artifacts in EEG MC signals. It has also been evaluated and compared with a supervised method (SM) to detect artifacts.

The UMED allows to find an optimal classification of the events. This classification can be applied in real-time, either training a SVM classifier or operating as a supervised method. The UMED has been mainly developed to analyze artifacts. For example, to capture the frequency and duration of artifacts for psycho-physiological studies. The UMED also provides, in an unsupervised way, the *NIC* required to remove artifacts with the help of ICA.

**Acknowledgments:** This work was supported in part by the Spanish Ministry of Economy and Competitiveness under Project DPI2011-27818-C02-02, DPI2014-55932-C2-2-R, ENE2015-64914-C3-1-R, ENE2015-64914-C3-2-R and FEDER funds.

**Author Contributions:** A.M. researched the literatures, designed the algorithm, analyzed the results and wrote the manuscript; R.D. researched the literatures, wrote the paper, supervised the overall study; J.V., N.D and S. D.-C. participated in revising the manuscript. All authors have approved the final manuscript.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Lance, B.J.; Kerick, S.E.; Ries, A.J.; Oie, K.S.; McDowell, K. Brain-computer interface technologies in the coming decades. *Proc. IEEE* **2012**, *100*, 1585–1599.
2. James, C.J.; Gibson, O.J. Temporally constrained ICA: An application to artifact rejection in electromagnetic brain signal analysis. *IEEE Trans. Biomed. Eng.* **2003**, *50*, 1108–1116.
3. Junghöfer, M.; Elbert, T.; Tucker, D.M.; Rockstroh, B. Statistical control of artifacts in dense array EEG/MEG studies. *Psychophysiology* **2000**, *37*, 523–532.
4. Nolan, H.; Whelan, R.; Reilly, R.B. FASTER: Fully automated statistical thresholding for EEG artifact rejection. *J. Neurosci. Methods* **2010**, *192*, 152–162.
5. Delorme, A.; Makeig, S. EEGLAB: An open source toolbox for analysis of single-trial EEG dynamics including independent component analysis. *J. Neurosci. Methods* **2004**, *134*, 9–21.
6. Oostenveld, R.; Fries, P.; Maris, E.; Schoffelen, J.-M. FieldTrip: Open source software for advanced analysis of MEG, EEG, and invasive electrophysiological data. *Comput. Intell. Neurosci.* **2011**, *2011*, 156869.
7. Durka, P.J.; Klekowicz, H.; Blinowska, K.J.; Szelenberger, W.; Niemcewicz, S. A simple system for detection of eeg artifacts in polysomnographic recordings. *IEEE Trans. Biomed. Eng.* **2003**, *50*, 526–528.
8. Lawhern, V.; Hairston, W.D.; Robbins, K. DETECT: A MATLAB toolbox for event detection and identification in time series, with applications to artifact detection in EEG signals. *PLoS ONE* **2013**, *8*, e62944.
9. Hido, S.; Tsuboi, Y.; Kashima, H.; Sugiyama, M.; Kanamori, T. Statistical outlier detection using direct density ratio estimation. *Knowl. Inf. Syst.* **2010**, *26*, 309–336.

10. Camci, F.; Chinnam, R.B. General support vector representation machine for one-class classification of non-stationary classes. *Pattern Recognit.* **2008**, *41*, 3021–3034.
11. Sadik, S.; Gruenwald, L. DBOD-DS: Distance based outlier detection for data streams. In Proceedings of 21st International Conference, DEXA 2010, Bilbao, Spain, 30 August–3 September 2010; pp. 122–136.
12. Takeuchi, J.; Yamanishi, K. A unifying framework for detecting outliers and change points from time series. *IEEE Trans. Knowl. Data Eng.* **2006**, *18*, 482–492.
13. O'Regan, S.; Faul, S.; Marnane, W. Automatic detection of EEG artefacts arising from head movements using EEG and gyroscope signals. *Med. Eng. Phys.* **2013**, *35*, 867–874.
14. Gardner, A.B.; Hall, H.; Krieger, A.M. One-class novelty detection for seizure analysis from intracranial EEG. *J. Mach. Learn. Res.* **2006**, *7*, 1025–1044.
15. Jung, T.P.; Makeig, S.; Humphries, C.; Lee, T.W.; McKeown, M.J.; Iragui, V.; Sejnowski, T.J. Removing electroencephalographic artifacts by blind source separation. *Psychophysiology* **2000**, *37*, 163–178.
16. Mannan, M.M.N.; Kim, S.; Jeong, M.Y.; Kamran, M.A. Hybrid EEG—Eye tracker: Automatic identification and removal of eye movement and blink artifacts from electroencephalographic signal. *Sensors* **2016**, *16*, doi:10.3390/s16020241.
17. Rokach, L.; Maimon, O. Chapter 15—Clustering methods. In *Data Mining and Knowledge Discovery Handbook*; Springer: New York, 2010.
18. Sokal, R.R.; Rohlf, F.J. The comparison of dendrograms by objective methods. *Taxon* **1962**, *11*, 33–40.
19. Halkidi, M.; Vazirgiannis, M. Clustering validity assessment: Finding the optimal partitioning of a data set. In Proceedings of the IEEE International Conference on Data Mining, San Jose, CA, USA, 2 December 2001; pp. 187–194.
20. Burg, J.P. Maximum entropy spectral analysis. In Proceedings of the 37th Annual International SEG Meeting, Soc. of Explor. Geophys, Oklahoma City, Okla, 31 October 1967.
21. Lawhern, V.; Hairston, W.D.; McDowell, M.; Westerfield, K.R. Detection and classification of subject-generated artifacts in EEG signals using autoregressive models. *Neurosci. Methods* **2012**, *208*, 181–189.
22. Jolliffe, I.T. Principal Component analysis. In *Encyclopedia of Statistics in Behavioral Science*, 2nd ed.; Springer: New York, 2002; Volume 30, p. 487.



23. Strehl, A.; Ghosh, J. Cluster ensembles—A knowledge reuse framework for combining multiple partitions. *J. Mach. Learn. Res.* **2002**, *3*, 583–617.
24. Melia, U.; Clariá, F.; Vallverdú, M.; Caminal, P. Filtering and thresholding the analytic signal envelope in order to improve peak and spike noise reduction in EEG signals. *Med. Eng. Phys.* **2014**, *36*, 547–553.
25. Mahajan, R.; Morshed, B. Unsupervised eye blink artifact denoising of EEG data with modified multiscale sample entropy, kurtosis and wavelet-ICA. *IEEE J. Biomed. Heal. Inform.* **2014**, *2194*, 1–8.
26. Daly, I.; Scherer, R.; Billinger, M.; Muller-Putz, G. FORCe: Fully online and automated artifact removal for brain-computer interfacing. *IEEE Trans. Neural Syst. Rehabil. Eng.* **2014**, *4320*, 1–13.
27. Nicolaou, N.; Nasuto, S.J. Automatic artefact removal from event-related potentials via clustering. *J. VLSI Signal Process. Syst. Signal Image Video Technol.* **2007**, *48*, 173–183.
28. Zeng, K.; Chen, D.; Ouyang, G.; Wang, L.; Liu, X.; Li, X. An EEMD-ICA approach to enhancing artifact rejection for noisy multivariate neural data. *IEEE Trans. Neural Syst. Rehabil. Eng.* **2015**, *4320*, doi:10.1109/TNSRE.2015.2496334.
29. Naseer, N.; Hong, K.S. fNIRS-based brain-computer interfaces: a review. *Frontiers in Human Neuroscience* **2015**, Volume 9, Article 3.
30. Coyle SM.; Ward TE; Markham CM. Brain-computer interface using a simplified functional near-infrared spectroscopy system. *J Neural Eng.* **2007**, *4*, 219–226.
31. Fazli S.; Mehnert J.; Steinbrink J.; Curio G.; Villringer A.; Muller KR.; Blankertz B. Enhanced performance by a hybrid NIRS-EEG brain computer interface. *NeuroImage* **2012**, *59*, 519–529.
32. Naseer, N.; Hong, K.S. Reduction of Delay in Detecting Initial Dips from Functional Near-Infrared Spectroscopy Signals Using Vector-Based Phase Analysis. *International Journal of Neural Systems* **2016**, Volume 26, Article 1.



## CAPÍTULO 5


### ART.2: UNSUPERVISED EVENT DETECTION AND CLASSIFICATION OF MULTICHANNEL SIGNALS

Resumen:


- *Revista:* Expert Systems with Applications
- *Current Impact Factor:* 2.981; *5-Year Impact Factor:* 2.879
- *Selección de la primera página del artículo publicado:*

Expert Systems With Applications 54 (2016) 294–303

Contents lists available at [ScienceDirect](#)

 **Expert Systems With Applications**  
An International Journal

Journal homepage: [www.elsevier.com/locate/eswa](http://www.elsevier.com/locate/eswa)

**Unsupervised event detection and classification of multichannel signals** 

Angel Mur<sup>a,\*</sup>, Raquel Dormido<sup>a</sup>, Jesús Vega<sup>b</sup>, Sebastian Dormido-Canto<sup>a</sup>, Natividad Duro<sup>a</sup>

<sup>a</sup>Department of Computer Sciences and Automatic Control, UNED, Juan del Rosal 16, 28040 Madrid, Spain  
<sup>b</sup>National Fusion Laboratory by Magnetic Confinement, CIEMAT, Complutense 40, 28040 Madrid, Spain

---

**ARTICLE INFO**

**Keywords:**  
Classification  
Event characterization  
Event detection  
Multichannel signal  
Temporal sequence

**ABSTRACT**

In this paper, we present a new unsupervised method to classify a set of Multichannel Signals (MC) with unknown events. Each signal is characterized by a sequence of events where the number of events, the start time and the duration between events can change randomly. The proposed method helps in the classification and event detection of the MC signals by an expert which usually becomes a tedious and difficult task. To this end, first, the problem of classification of MC signals characterized by a succession of events is analyzed by transforming the MC signals into a set of temporal sequences of easy interpretation. The algorithm detects events by means of an optimal unsupervised classification. It is not necessary to know the nature of the events and formulate hypotheses regarding their behavior. Then, a set of multichannel electromyographic (EMG) signals with events is generated. These MC signals are used to test the proposed method.

© 2016 Elsevier Ltd. All rights reserved.

# Unsupervised Event Detection and Classification of Multichannel Signals

Angel Mur<sup>a</sup>, Raquel Dormido<sup>a</sup>, Jesús Vega<sup>b</sup>, Sebastian Dormido-Canto<sup>a</sup>, Natividad Duro<sup>a</sup>

<sup>a</sup> Department of Computer Sciences and Automatic Control, UNED, Juan del Rosal 16 - 28040 Madrid, Spain, a.r.m.g@outlook.fr, raquel@dia.uned.es, nduro@dia.uned.es, sebas@dia.uned.es

<sup>b</sup> National Fusion Laboratory by Magnetic Confinement. CIEMAT, Complutense 40 - 28040 Madrid, Spain, jesus.vega@ciemat.es

## ABSTRACT

In this paper, we present a new unsupervised method to classify a set of Multichannel Signals (*MC*) with unknown events. Each signal is characterized by a sequence of events where the number of events, the start time and the duration between events can change randomly. The proposed method helps in the classification and event detection of the *MC* signals by an expert which usually becomes a tedious and difficult task. To this end, first, the problem of classification of *MC* signals characterized by a succession of events is analyzed by transforming the *MC* signals into a set of temporal sequences of easy interpretation. The algorithm detects events by means of an optimal unsupervised classification. It is not necessary to know the nature of the events and formulate hypotheses regarding their behavior. Then, a set of multichannel electromyographic (*EMG*) signals with events is generated. These *MC* signals are used to test the proposed method.

**Keywords:** Classification, Event Characterization, Event Detection, Multichannel Signal, Temporal Sequence.

## 1. Introduction

Temporal sequence refers to a sequence of happenings or events in a time interval. An event is something that takes place at a time  $t_E$  and it reflects some kind of change in a temporal evolution. An important characteristic of temporal sequences is the order in which events take place in time. Other important characteristic is the duration  $d_E$  between  $t_E$  and the next event.

A temporal sequence is normally represented by a series of nominal symbols from a particular alphabet. Every event is characterized by a symbol. In this way, an event with symbol  $E$  is described by 2 elements  $(E, t_E)$ . The 3 elements  $(E, t_E, d_E)$  represent a state.

A temporal sequence  $S$  made of  $u$  events  $E^i$  for  $i=1, \dots, u$  can be described by a  $u$ -tuple of states  $S = \langle (E^1, t_E^1, d_E^1), (E^2, t_E^2, d_E^2), \dots, (E^u, t_E^u, d_E^u) \rangle$  or  $S = \langle (E^1, t_E^1), (E^2, t_E^2), \dots, (E^u, t_E^u) \rangle$ . For simplicity, we will use  $S = \langle E^1, E^2, \dots, E^u \rangle$ .

Temporal sequences appear in different domains such as engineering, medicine and finance among others (Mitsa, 2010).

The Web usage can also be analyzed by temporal sequences where events are related to the pages accessed by the users as well as the time and duration of visits. Other example could be the sequence of activities (eating, walking...) that people carry out for a specific period of time. Both examples show temporal sequences whose events can be found easily through observation.

In general, the number and temporal locations of events can be unknown within a temporal sequence. This means that events have to be recognized and located inside the sequence. It occurs, for example, in the monitoring of a physical system with  $Q$  sensors for a period of time  $[T_1, T_2]$ . In this case, every sensor represents a channel and simultaneously provides a signal describing part of the whole information. Signals of this type that are generated by multiple sensors are called Multichannel (*MC*) signals. In other words, a *MC* signal refers to a set of signals that show cross-channel similarity or correlation.

A *MC* signal with  $Q$  channels is represented by a vector  $X(t)=[C_1(t) C_2(t) \dots C_Q(t)]$  where  $C_q(t)$  is the signal of the channel  $q$  for  $q=1,2,\dots,Q$ .

A *MC* signal  $X(t)$  with events can also be described, in compact form, as a temporal sequence  $S$ .

In this paper, we are interested in classifying a set of different *MC* signals in which not only the number of events is variable but also their start time  $t_E$  and duration  $d_E$  between events are unknown and they can change randomly. This kind of classification is important in domains such as Bioengineering, Volcanology, Geophysics, Nuclear Fusion etc. where *MC* signals are present. The classification criterion used in this paper establishes that two *MC* signals belong to the same group if they have the same behavior for a period of time  $[T_1, T_2]$ . Two *MC* signals have the same behavior if they present the same events in the same order. The start time  $t_E$  of each event and the duration  $d_E$  between events may be different.

For example, if for the interval [12:00 p.m, 14:00 p.m] a person  $H_1$  walks 1h, cooks for 20 min., eats for 20 min., sleeps 20 min., other person  $H_2$  sleeps 30 min., cooks for 30min., walks 30 min., eats for 30 min. other person  $H_3$  walks 30 min., cooks for 15 min., eats for 25 min., sleeps 50 min. etc. In this example and according to the criterion mentioned above, only  $H_1$  and  $H_3$  have the same behavior.

Classification techniques are one of the most important domains in data mining. There are two types of classification procedures: supervised classification and unsupervised classification (Mitsa, 2010).

An unsupervised method can not classify a set of *MC* signals with unknown events (using the same temporal interval), due to the random nature of  $t_E$  and  $d_E$ . This method selects a feature vector (a  $\eta$ -dimensional vector of numbers) and the *MC* signals with a similar feature vector are classified into the same group. But, when  $t_E$  and  $d_E$  change randomly, in general, the feature vectors found for the *MC*

signals with the same behavior are significantly different and consequently these *MC* signals will not be classified together.

In *MC* signals with unknown events, we do not know the number, order, start time, duration and name of the events. As the events are not known, any classification method can not address the problem in a supervised way since it is necessary a prior knowledge of the events characteristics. Furthermore, a supervised method using known events is not able to detect when new events appear and they have not been considered a priori.

In this paper, we propose a new unsupervised method able to classify a set of *MC* signals with unknown events. To this end, first events are detected, then *MC* signals are transformed into a set of  $n$  sequences and finally the classification is performed.

To validate the proposed method a set of *EMG MC* signals with known events is used. The method detects events in these *EMG MC* signals by considering they are unknown. The good results obtained allow us to extend the use of this method to other type of *MC* signals with unknown events.

References (Antunes & Oliveira, 2001; Mitsa, 2010; Mohd. Shahnawaz, Ashish Ranjan, 2011; Y Leela Sandhya Rani, P Naga Deepthi, 2013) provide a survey of the most significant techniques to deal with temporal sequences, in particular temporal sequences classification.

Ref. (Xing, Pei, & Keogh, 2010) presents a review about sequence classification: feature based methods (Aggarwal, 2002; Ji, Bailey, & Dong, 2007; Littlestone, 1988; Ye & Keogh, 2009), sequence distance based methods (Kaján et al., 2006; Needleman & Wunsch, 1970; Xi, Keogh, Shelton, Wei, & Ratanamahatana, 2006), model based methods (Cheng, Carbonell, & Klein-Seetharaman, 2005; R. Durbin, S. R. Eddy, A. Krogh, 1998; Srivastava, Desai, Nandi, & Lynn, 2007; Yakhnenko, Silvescu, & Honavar, 2005), early classification on sequences (J. J. R. Diez, C. A. González, 2001), (Z. Xing, J. Pei, G. Dong, 2008) and semi-supervised learning on sequences (Weston et al., 2005).

Most of the work on event detection focuses on the change detection in data streams. It is based on certain windows to define and study changes in distributions (S. Muthukrishnan, Eric van den Berg, 2007). References (Cormode & Muthukrishnan, 2004; Gehrke, Ganti, Gehrke, & Ramakrishnan, 1999; Kifer, Ben-david, & Gehrke, 2004; Krishnamurthy, Sen, Zhang, & Chen, 2003) use two windows with a fixed size and estimate the change of distribution between them. References (Bartlett, Ben-David, & Kulkarni, 2000; Gama, Medas, Castillo, & Rodrigues, 2004; Widmer & Kubat, 1996) use windows of varied sized but this method is computationally expensive. Ref. (Ho, 2005) proposes a martingale framework for change detection where an adaptive window is used. Ref. (Bifet, Gavaldà, & Gavaldà, 2007) uses exponential histograms to detect change of various scales. Ref. (S. Muthukrishnan, Eric van den Berg, 2007) detects changes without a window specification. It provides a new sequential change detection

algorithm that improves the use of the Sequential Probability Ratio Test (Wald, 2004) to decide a change. Hidden Markov Models (*HMM*) (Dias & Ramos, 2014; Li, Fang, & Huang, 2015; Rabiner, 1989) are capable of modeling sudden changes in a signal as well as segment the signal into informative states. They can also classify signals in a supervised and unsupervised way. The main drawback of *HMM* is that they are models and consequently their accuracy and capacity of being generic is affected.

In our work, event detection and classification are related. Unsupervised classifications are used to detect and characterize events on a set of *MC* signals and subsequently, the events characterization allows classifying the *MC* signals.

In section 2, the problem of classification of *MC* signals in which there are events whose start time  $t_E$  and duration  $d_E$  can change randomly is analyzed. In section 3, we present the new method to detect events and classify the *MC* signals. In section 4, a set of *EMG MC* signals made of events with random  $t_E$  and  $d_E$  is generated. Then, the proposed method is tested using these signals. Finally in sections 5 and 6, a discussion and the conclusions of the paper are respectively presented.

## 2. Analysis of the classification of multichannel signals with events

A classic method of unsupervised classification (*CC*) for *MC* signals is built on the following steps: 1) the selection of a working window with significant duration, 2) the selection of a feature vector for each *MC* signal on that window, and 3) the application of an unsupervised classification method to the feature vectors chosen.

In this section the problem of classification of temporal sequences with unknown events is analyzed. As a particular case this analysis is valid for *MC* signals. The classification criterion of the *MC* signals mentioned above consists of grouping *MC* signals that show the same behavior in an interval  $[T_1, T_2]$ . For example, Fig. 1 shows two temporal sequences  $S_1$  and  $S_2$  with four events  $A, B, C$  and  $D$ . Both sequences have the same behavior (because they present the same events in the same order) and consequently,  $S_1$  and  $S_2$  are classified into the same group.

In Fig. 2, there are four temporal sequences  $S_1, S_2, S_3, S_4$  with four events  $A, B, C, D$  in an interval  $[T_1, T_2]$ . According to the defined criterion, the temporal sequences are classified into two groups  $G_1=\{S_1, S_3\}$  and  $G_2=\{S_2, S_4\}$ . The duration and the start time of the events of each group is the same. And therefore, a *CC* applied to the window  $[T_1, T_2]$  would coincide with the result of our criterion, since the feature vectors are similar in each group.

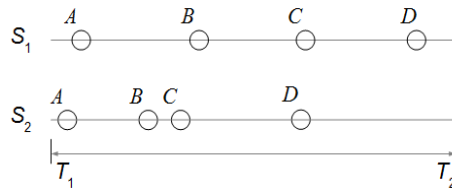


Fig. 1 Two temporal sequences with 4 events and the same behavior.

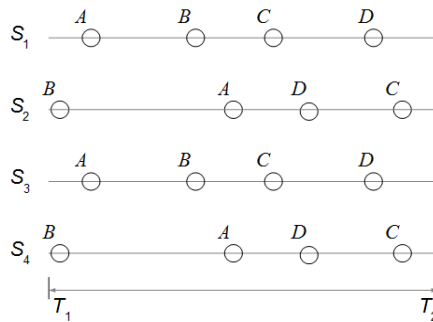


Fig. 2 Four temporal sequences and 2 groups with the same behavior. Each group has events with the same start point and duration.

A different situation is shown in Fig.3. In this case, there are four temporal sequences  $S_1, S_2, S_3, S_4$  with four events  $A, B, C, D$  in an interval  $[T_1, T_2]$ . According to the defined criterion, the temporal sequences are classified into two groups  $G_1=\{S_1, S_3\}$  and  $G_2=\{S_2, S_4\}$ . However, for this example, the duration and start time of the events of each group are different and a  $CC$  applied to the window  $[T_1, T_2]$  will no match the result of our criterion. That is because of the random start time and duration between two consecutive events. In this case the feature vectors are different in each group.

As in the example shown in Fig. 3, we could have a set of  $MC$  signals in which the possible succession of events and their characteristics (number, order, start time, duration and name) are unknown. Therefore, it is necessary to develop a new method capable of detecting events and classifying  $MC$  signals according to the criterion previously defined.

In the next section, it will be shown the new method to detect events and classify the  $MC$  signals.



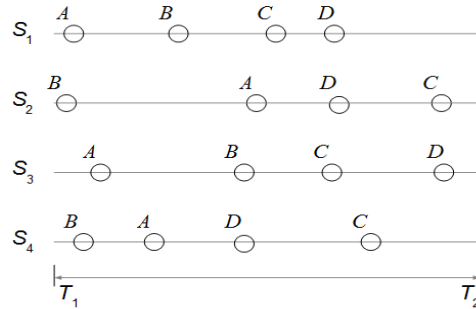


Fig. 3 Four temporal sequences and 2 groups with the same behavior. Each group has events with different start point and duration.

### 3. Method of detection and classification

This method makes use of 4 concepts: the Hierarchical clustering (*HC*) algorithm (Rokach & Maimon, 2005), the Cophenetic Correlation Coefficient *CCC* (Sokal & Rohlf, 1962), the *S\_Dbw* Validity Index (Halkidi & Vazirgiannis, 2001) and the Silhouette Validity Index (Rousseeuw, 1987).

Hierarchical clustering (*HC*) groups data over a variety of scales by creating a cluster tree or dendrogram following these steps: 1) Find the similarity or dissimilarity between every pair of objects in the data set 2) Group the objects into a binary, hierarchical cluster tree (linkage) 3) Determine where to cut the hierarchical tree into clusters.

The Cophenetic Correlation Coefficient *CCC* measures the correlation between the distance values calculated during tree building and the observed distance. The *CCC* is a measure of how faithfully a dendrogram maintains the original pairwise distances. The magnitude of *CCC* should be very close to 1 to achieve a high-quality solution of the *HC* algorithm.

The *S\_Dbw* Validity Index is an index used for measuring the “goodness” of a clustering result. Its definition is based on cluster compactness and separation but it also takes into consideration the density of the clusters. Lower index value indicates better clustering schema.

Another well-known index to validate a clustering partition is the Silhouette index. It is based on geometrical considerations combining both cohesion and separation of the clusters. The silhouette value for each point is a measure of how similar that point is to points in its own cluster compared to points in other clusters, and ranges from -1 to +1. The average Silhouette is an overall measure of the goodness of a clustering. The higher value, the better. This index is a robust measure for noisy data.

The proposed method (*UMEDC*) classifies a set of *MC* signals with unknown events by transforming the *MC* signals into a set of temporal sequences made of meaningful events. The temporal sequences are represented by strings and/or feature vectors of the same size. Their pairwise similarity allows to carry out an unsupervised classification of the *MC* signals.

The method selects a set of windows by means of an sliding window that moves along each *MC* signal. Each window is characterized by a feature vector that picks up the behavior of the *MC* signal. The detection of the events in each *MC* signal is carried out by means of an unsupervised classification of its feature vectors. The quality of this classification (and the number of events detected) depends significantly on the size of the sliding window. This is why the algorithm searches for an optimal window size that provides the maximum compactness and separation of the content between the events.

Given a set of *MC* signals with unknown events  $X_j$  ( $j=1\dots n$ ) to be classified for a period of time  $[T_1, T_2]$ , the steps of *UMEDC* can be summarized as follows:

**Step 1: Selection of windows.** For each *MC* signal  $X_j$  ( $j=1\dots n$ ), the algorithm uses a sliding window approach. Fig. 4 shows the details of the sliding window in a *MC* signal  $X$  along the interval  $[T_1, T_2]$ . If the window  $W_i$  of size  $L_w$  is defined by the interval  $[t_1, t_2]$  then the window  $W_{i+1}$  is defined by  $[t_1+d, t_2+d]$  with  $d>0$ . As first step a set of *IN* windows is selected for a specific  $L_w$  and  $d$ .

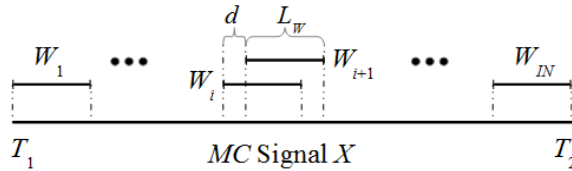


Fig. 4 Different windows along the *MC* signal  $X$ .

**Step 2: Feature matrix (*FV*) calculation.** In this step, for each *MC* signal  $X_j$  ( $j=1\dots n$ ), a matrix  $(FV)_j$  of *IN* rows is generated. Each row  $i$  of  $(FV)_j$  represents a  $Q$ -feature vector of the window  $(W_i)_j$ . So first we calculate for each channel  $q$  and window  $(W_i)_j$  a feature vector of size  $s$ . Then, these features vectors are concatenated to form a unique  $Q$ -feature vector  $(Fv_i)_j$  (with  $S = Q \times s$  features) for each  $(W_i)_j$ . At last, all the feature vectors  $(Fv_i)_j$  (for  $i=1\dots IN$ ) are saved in a matrix  $(FV)_j$  of size  $IN \times S$  concatenating them vertically (see Fig. 5).

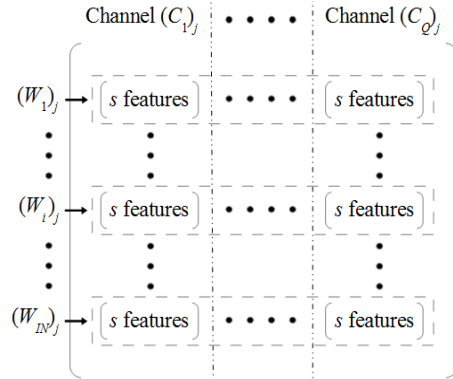


Fig. 5 Matrix  $(FV)_j$  of the MC Signal  $X_j$

Two autoregressive coefficients are good candidates to form the feature vector for each channel  $q$  and window  $W$ . The coefficients  $a_1$  and  $a_2$  of an autoregressive model  $AR(2)$  of a window  $W(m)$  for a single channel can be written as:

$$W(m) = \sum_{i=1}^2 a_i W(m-i) + e(m) \quad (1)$$

where  $e(m)$  is zero-mean white noise. The  $a_i$  coefficients can be estimated using Burg's method (Burg, 1975). Normally the  $Q$ -feature vector with a  $AR(2)$  (of length  $Q \times 2$ ) offers superior or equal performance than a  $AR(p)$  with  $p > 2$ . This has been proven by detecting artifacts in *EEG* recordings (Lawhern, Hairston, & Robbins, 2013; V. Lawhern, WD. Hairston, K. McDowell, M. Westerfield, 2012). And it is computationally less intensive.

**Step 3: Selection of a window size  $L_w$  using the CCC.** First, for each  $X_j$  ( $j=1 \dots n$ ), the  $(CCC)_j$  of the  $IN$  rows of the matrix  $(FV)_j$  is calculated. If  $(CCC)_j$  reaches approximately a particular threshold  $U$ , it is selected the value of  $(L_w)_j$ . This value is named  $(L_w^0)_j$  and its feature matrix is  $(FV^0)_j$ . In case  $(CCC)_j$  is much lower than  $U$ , we go back to the step 1 to choose a bigger  $(L_w)_j$  and then, a new  $(FV)_j$  and  $(CCC)_j$  are tested.

The  $(CCC)_j$  threshold  $U$  is defined as 0,85. This value guarantees that the *HC* algorithm will find a significant classification. From a practical point of view, this threshold is used to reduce the grid search for the window length in step 4. For small window sizes (some dozens of samples) the  $CCC$  value can be  $< 0,85$ . In this case the  $CCC$  is used to determine a minimum window size from which the step 4 will start. In step 4, the  $S\_dbw$  takes the responsibility to calculate an

optimum window size. A low value of  $S\_dbw$  comes with a high  $CCC$  value. However, the  $CCC$  is not defined from compactness and separation of groups and it can not be used to determine an optimal group number.

For each window size the processing time of  $CCC$  is low. So a possible strategy to determine  $(L_w^0)_j$  is to select a small group of window sizes and check if the  $CCC > 0.85$  is respected from a particular window size.

**Step 4: Selection of an optimal window size  $L_w^H$  and its unsupervised classification.** First, for each  $MC$  signal  $X_j$  ( $j=1\dots n$ ), the size of the window is increased  $N_L$  times by means of  $(L_w^m)_j = (L_w^0)_j + m \times D_L$  for  $m = 0\dots N_L$  and the integer  $D_L \geq 1$ . Second for each  $(L_w^m)_j$ , it is calculated its feature matrix  $(FV^m)_j$ . Third, the  $HC$  algorithm is applied  $N_G$  times to classify the rows of each matrix  $(FV^m)_j$  in different partitions i.e. in 2, 3, ...,  $N_G+1$  groups. Fourth, for each partition, its  $(S\_Dbw^m_g)_j$  index (with  $g=2, \dots, N_G+1$ ) is calculated. The minimum value of the index, named  $(Min^m)_j$ , determines the optimal number of groups  $(No^m)_j$ . The minimum value  $(Min^H)_j$  between the  $(Min^m)_j$  determines the optimal window size  $(L_w^H)_j$ , the optimal feature matrix  $(FV^H)_j$  and the optimal number of groups  $(No^H)_j$ .

The selected  $N_L$  value allows to check a significant number of window sizes. Each  $MC$  signal  $X_j$  has an initial period of time ( $IPT$ ) where there are no events. If  $IPT$  is known, then  $(L_w^{NL})_j$  has to be  $< IPT$ . If  $IPT$  is not known, then it is possible to add at the start of  $X_j$  a synthetic portion of signal without events. When the size  $L_w^H$  is near to  $L_w^{NL}$  a bigger value of  $N_L$  is chosen (with the help of a synthetic  $IPT$ , if necessary) to ensure that  $L_w^H$  is well surrounded by other suboptimal sizes.

The selected  $N_G$  value allows to check a significant number of partitions. In case the optimal number  $(No^m)_j$  is near to  $N_G$ , a bigger value of  $N_G$  is chosen to ensure that  $(No^m)_j$  is well surrounded by other suboptimal group numbers.

**Step 5: Event Detection by means of the optimal unsupervised classification.** For each  $MC$  signal  $X_j$  ( $j=1\dots n$ ), it is accomplished the unsupervised classification for the predetermined number of groups  $(No^H)_j$ . The condition  $(CCC)_j > U$  and the selection of  $(Min^H)_j$  allow the  $HC$  algorithm to find a high-quality classification with the best group compactness and separation taking into consideration the density of the clusters.

The events can be detected along their  $MC$  signal. These are placed at the beginning and the end of each group of consecutive intervals  $[(W_i)_j, (W_{i+1})_j, \dots, (W_g)_j]$  where  $(1 \leq i \leq g \leq IN)$  with the same group number. The beginning and the end can be defined using the next formula (Lawhern et al., 2013):

$$[(i-1) \times d + M \times L_w - \frac{d}{2}, (g-1) \times d + M \times L_w + \frac{d}{2}] \quad (2)$$

where  $i$  is the number of the first interval  $W_i$ ,  $g$  is the number of the last interval  $W_g$ ,  $d$  is the slide width and  $M = 6/7$ . The product  $L_w \times 6/7$  is more accurate than using the midpoint with  $M = 1/2$ . (2) uses only  $d$  samples per interval along the

signal without overlapping. In this way, intervals between events are uniquely labeled regardless of whether events are placed in the intersection of two overlapping windows classified in a different way.

This formula is used to plot the events. This graphical representation is important because it allows to evaluate the event detection in a qualitative manner.

**Step 6: Global event characterization.** For each *MC* signal, it is calculated the centroids of the feature vectors of the intervals between events. Then, an unsupervised classification (*UC*) of the centroids (of all *MC* signals) using *HC* and the *S\_Dbw* index provides the group numbers for these centroids that characterize the events in a global way. For each *MC* signal  $X_j$  ( $j=1\dots n$ ), the group numbers that characterize the events form a temporal sequence  $S_j$  ( $j=1\dots n$ ). Then, the  $n$  temporal sequences  $S_1, \dots, S_n$  are represented by  $n$  strings  $ST_1, \dots, ST_n$  formed by concatenating the group numbers obtained in the *UC*.

For each  $S_j$ , if two or more consecutive events are characterized with the same group number we fuse them in a single group number and the two-digit group numbers are renamed using the alphabet consecutively (Ex:  $10 \rightarrow a$ ,  $11 \rightarrow b$  and so on). Once the group numbers has been fused and renamed for each  $S_j$ , we construct a string  $ST_j$  that shows the temporal sequence in a compact way (see Table 1).

At the end of the step 6, we have transformed the  $n$  *MC* signals into  $n$  temporal sequences  $S_1, \dots, S_n$  represented by  $n$  strings  $ST_1, \dots, ST_n$ .

**Table 1**

Process to Transform a *MC* Signal with Events  $X$  into a Temporal Sequence

<i>MC</i> signal	$X$
Events	$\rightarrow E^1 \rightarrow E^2 \rightarrow E^3 \rightarrow$
Group Numbers	$2 \rightarrow 3 \rightarrow 3 \rightarrow 10$
Fusion and renaming	$2 \rightarrow 3 \rightarrow a$
String of the temporal sequence	$23a$

**Step 7: Events Filtering.** Some events can be filtered under a specific criterion (depending on the desired classification) and as a result the new sequences  $NS_1, \dots, NS_n$  contain fewer characters. Their strings are named  $NST_1, \dots, NST_n$ .

We can define the Total Number of Intervals (*TNI*) per group using all the intervals (characterized in Step 6) from all the signals. In order to filter out insignificant events, the events with a *TNI* less than a predefined ratio can be discarded.

This filtering process can be interesting in some kind of *MC* signal to get a simpler sequence of events that characterize better its behavior.

**Step 8: Calculate the distance array  $D_S$  of the MC signals.**

*First option ( $D_S=D_{S1}$ ):* The distance array  $D_{S1}$  contains the pairwise distances between the  $n$  strings  $ST_1, \dots, ST_n$  (or  $NST_1, \dots, NST_n$  after the step 7).

The pairwise distances are calculated using their pairwise similarities. We measure their pairwise similarities by the  $S^3M$  similarity function (K. Santhisree1, A. Damodaram, 2010; P. Kumar, P. Radha Krishna, 2010). The  $S^3M$  of the strings  $A$  and  $B$  is given by

$$S^3M(A, B) = \frac{p*LLCS(A, B)}{\max(|A|, |B|)} + \frac{q*|A \cap B|}{|A \cup B|} \quad (3)$$

where  $p + q = 1$  and  $p, q \geq 0$ . It has been selected  $p = q = 0,5$ . The  $LLCS$  is the length of the longest common string.

The distance between  $A$  and  $B$  is the following:

$$d_s(A, B) = 1 - S^3M(A, B) \quad (4)$$

All the pairwise distances of the  $ST_1, \dots, ST_n$  (or  $NST_1, \dots, NST_n$ ) using (4) are stored in an array  $D_{S1}$ .

*Second option ( $D_S=D_{S2}$ ):* The temporal sequences  $S_1, \dots, S_n$  (or  $NS_1, \dots, NS_n$  after the step 7) are transformed into feature vectors  $FS_1, \dots, FS_n$ , of the same size by means of a Hidden Markov Model (Rabiner, 1989). The distance array  $D_{S2}$  contains the pairwise distances between the  $n$  feature vectors  $FS_1, \dots, FS_n$ .

A *HMM* depends on a set of  $NH$  hidden states  $\{H_1, \dots, H_{NH}\}$  and a set  $V$  of  $NV$  symbols. The set  $V$  contains the alphabet obtained in step 6 or 7.

A Hidden Markov Model  $\lambda$  is represented by a triplet,  $\lambda = (A, B, \pi)$ . The transition matrix  $A = \{a_{ij} = P[q_{t+1} = H_j | q_t = H_i], 1 \leq i, j \leq NH\}$  represents the probability of moving from state  $H_i$  to state  $H_j$ , with  $a_{ij} \geq 0$  and the sum of each row is 1. The  $q_t$  denotes the state occupied by the model at time  $t$ . The emission matrix  $B = \{b(o|H_j)\}$ , indicates the probability of emission of symbol  $o \in V$  when system state is  $H_j$ . The initial state probability distribution is  $\pi = \{\pi_i\}$  with  $\pi_i = P[q_1 = H_i], \pi_i \geq 0$  and  $1 \leq i \leq NH$ . The sum of  $\pi_i$  is 1.

First, for a specific  $NH$  and  $NV$ , the matrices  $A$ ,  $B$  and  $\pi$  are randomly initialized. Then, with the help of the Baum-Welch algorithm (Rabiner, 1989), a model  $\lambda_{NH}$  is trained using the sequences  $S_1, \dots, S_n$  (or  $NS_1, \dots, NS_n$ ) and finally for each sequence  $S_i$ , the log-likelihood  $\log P(S_i | \lambda_{NH})$  of the sequence with respect to the *HMM* model is evaluated. In this way, each sequence  $S_i$  is represented by a feature vector  $FS_i$  of one parameter. It is possible to get more parameters per feature vector using the same procedure with different values of  $NH$ . Because the training algorithm converges to a local maximum, the feature vectors are dependent on the initial model from which the training algorithm starts.

All the pairwise distances of  $FS_1, \dots, FS_n$ , calculated by using an euclidean metric, are stored in an array  $D_{S2}$ .

**Step 9: Automatic Classification of the MC signals.** In a general case we could have a large number of signals to classify. Furthermore, their temporal sequences could have a large number of symbols and the signals with the same behavior could also have similar temporal sequences but not identical. Therefore the  $n$  MC signals  $X_j$  ( $j=1\dots n$ ) should be classified automatically instead of trying to compare their sequences or strings (obtained in steps 6 and 7) manually. Two types of solutions have been considered:

*Solution 1:* The  $n$  sequences  $NS_1, \dots, NT_n$  (or strings  $NST_1, \dots, NST_n$ ) are classified using an unsupervised method able to classify from their pairwise distances ( $D_S$ ). This unsupervised classification can be performed using a HC algorithm along with the Silhouette validity index to calculate the optimal number of groups  $N_O$ .

A Hierarchical Clustering (HC) classifies the sequences  $NS_1, \dots, NT_n$  (or strings  $NST_1, \dots, NST_n$ ) using  $D_S$ . First, the classification is performed for different group numbers from  $k = 2$  to  $k^*$ . These classifications are stored in the columns of a matrix  $M_{HC}$  (with size  $n \times (k^*-1)$ ). Then, when an optimal number of clusters  $N_O$  is calculated, the classification of the strings (or sequences) concludes choosing the column  $N_O-1$  of  $M_{HC}$  as the optimal classification.

To calculate  $N_O$ , the Silhouette index makes use of the matrix  $M_{HC}$  and the distances between strings (or sequences) (saved in  $D_S$ ). In contrast to other validity index, Silhouette does not need to calculate the centroids of the clusters. Therefore, this index is suitable for a set of sequences (or strings).

This solution is for  $D_S=D_{S1}$  or  $D_S=D_{S2}$ .

*Solution 2:* Other way to calculate the classification of  $NST_1, \dots, NST_n$  and  $N_O$  from  $D_S$  consists on using the Spectral Clustering Algorithm (SC) (Ng, Jordan, & Weiss, 2001). The SC treats the data clustering as a graph partitioning problem without make any assumption on the form of the data clusters. It forms the associated Laplacian matrix and compute its eigenvalues and eigenvectors. Then, it maps each data point to a lower-dimensional representation based on two or more eigenvectors. In order to get  $k$  clusters, it is selected the  $k$  largest eigenvectors (eigenvectors whose eigenvalues are the largest in magnitude) to construct a matrix  $U_k$ . The row  $j$  of  $U_k$  represents the string  $NST_j$ .

The normalized Laplacian  $L_{sym}$  is defined as  $L_{sym} = Q^{-1/2} L Q^{-1/2}$  where  $L$  form the affinity matrix:

$$L(i, j) = e^{-\frac{d^2(NST_i, NST_j)}{2\sigma^2}} \quad \text{for } i \neq j \quad \text{and } L(i, i) = 0 \quad (5)$$

The matrix  $Q$  is a diagonal matrix whose  $(i, i)$ -element is the sum of  $L$ 's  $i$ -th row. The SC uses a scaling parameter  $\sigma$ .

For each matrix  $U_k$ , it is calculated the distances between all the pair of rows. These distances can be saved in arrays  $D_k^U$ . Then, we calculate the correlation  $C_k$  between each  $D_k^U$  and  $D_S$ . It is selected the matrix  $U^H$  with the highest correlation

$C^H$ . The closer  $C^H$  is to 1, the more  $U^H$  represents the  $n$  strings  $NST_1, \dots, NST_n$ . Via the rows of  $U^H$  the  $n$  strings are transformed into  $n$  feature vectors of the same size.

Once  $U^H$  is known, it is possible to use, for example Silhouette, to determine  $N_o$  using the matrix  $M_{HC}$  (described above) and the matrix  $U^H$ .

This solution is more sophisticated than the first one. But, once the groups of the classification have been obtained, it has the advantage of being able to calculate the centroids of each group using  $U^H$ . The nearest strings of the centroids are the most representative ones of the groups. And consequently, it is possible to choose the most representative  $MC$  signal per group.

This solution is only for  $D_S=D_{S1}$  because  $D_{S2}$  was already obtained from a set of feature vectors.

#### 4. Testing the algorithm

##### 4.1 EMG multichannel signal with events

Electromyography (*EMG*) is a technique for evaluating and recording the electrical activity produced by skeletal muscles. We have generated 20 *EMG MC* signals with events (from the right forearm) which will be used to test in the next section the new method developed to detect events and classify signals. They are denoted as  $X_j, j=1 \dots 20$ .

Although the proposed method works for any number of events, in our particular case, we are considering that each channel presents a maximum of 3 hand movements (Hand Open, Hand Close, Wrist Flexion) without repetition ( $u \leq 3$ ) for 10 seconds beginning with the hand in rest state. The subject holds each hand movement for a random duration between 800 ms and 3 seconds. The order of these movements was randomized. The *EMG* signals are formed using the data from the Myoelectric Control Development toolbox available at (Chan, n.d.). The data acquisition process is described in (Chan ADC, 2007). The data were downsampled from 3000 Hz to 1000 Hz.

Each *EMG MC* signal  $X_j$  is characterized by eight channels ( $Q=8$ ) (see Fig. 6).

The hand movements for the 20 *EMG MC* signals are shown in Table 2 where  $O$  is Hand Open,  $C$  is Hand Close,  $F$  is Wrist Flexion and  $R$  is Rest State. Within square brackets their start point in ms.

We want to classify the *EMG* signals according to their succession of hand positions.

In the next subsection, the new method able to classify the *MC* signals using the set of *EMG* signals of Table 2 is tested.

The first channel  $C_1(t)$  of three *EMG MC* signals are shown in the Fig. 7.



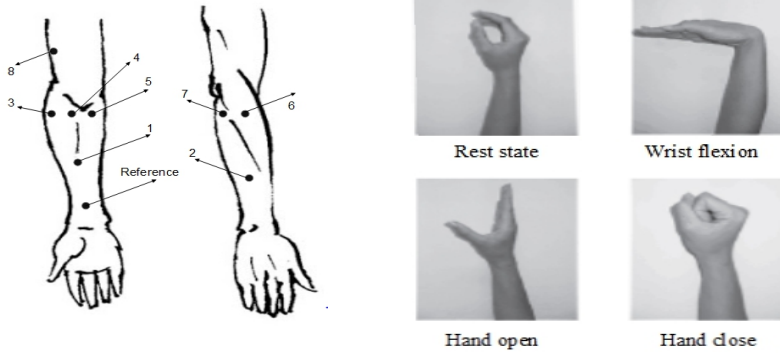


Fig. 6 The electrode placements on the right forearm and the hand movements selected.

**Table 2**  
Hand Movements for 10 Seconds

$X_1$	$R \rightarrow F[777] \rightarrow O[3344] \rightarrow C[6141]$	$X_{11}$	$R \rightarrow O[4262 \text{ ms}]$
$X_2$	$R \rightarrow F[8824]$	$X_{12}$	$R \rightarrow C[3478] \rightarrow F[4320] \rightarrow O[6852]$
$X_3$	$R \rightarrow F[4793] \rightarrow O[6447] \rightarrow C[7081]$	$X_{13}$	$R \rightarrow F[5439] \rightarrow O[6342] \rightarrow C[8568]$
$X_4$	$R \rightarrow O[9123]$	$X_{14}$	$R \rightarrow C[4791] \rightarrow F[7471]$
$X_5$	$R \rightarrow F[5903] \rightarrow C[7319] \rightarrow O[8038]$	$X_{15}$	$R \rightarrow C[3604] \rightarrow F[5192] \rightarrow O[7775]$
$X_6$	$R \rightarrow F[856] \rightarrow C[3392] \rightarrow O[6315]$	$X_{16}$	$R \rightarrow F[4083] \rightarrow C[6622] \rightarrow O[8414]$
$X_7$	$R \rightarrow F[7433]$	$X_{17}$	$R \rightarrow O[6415] \rightarrow F[7375]$
$X_8$	$R \rightarrow O[7932]$	$X_{18}$	$R \rightarrow C[9141]$
$X_9$	$R \rightarrow F[1642] \rightarrow O[6358] \rightarrow C[8383]$	$X_{19}$	$R \rightarrow O[6340] \rightarrow C[7709]$
$X_{10}$	$R \rightarrow C[7564]$	$X_{20}$	$R \rightarrow O[3568] \rightarrow C[6667] \rightarrow F[9103]$

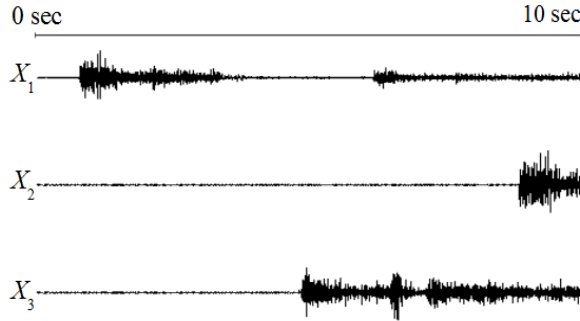


Fig. 7 The first channel of the *EMG* signals  $X_1, X_2, X_3$ .

#### 4.2 Events detection and classification of the *EMG MC* signals

In this subsection the proposed method is tested using the set of 20 *EMG MC* signals described in the previous subsection.

First, we apply the steps 1 to 5 using a slide width  $d=32$  samples and an autoregressive model  $AR(2)$  per window. In order to find the best window size  $L_w^H$ , it has been checked different window size from 200 to 250 samples ( $N_L=50$  and  $D_L=1$ ). These windows respects the threshold  $U = 0,85$ . Table 3 shows the best window size  $L_w^H$  and the optimal number of groups  $No^H$  per *MC* signal.

For each *MC* signal, the events can be located using the formula (2). Fig 8 shows the events found for three *EMG* signals.

Then, an unsupervised classification of the centroids by means of *HC* and the index  $S\_Dbw$ , as described in step 6, provides the global characterization of the events. The optimal number of groups of the centroids is 10. And the set of  $n$  *EMG* signals  $X_1, \dots, X_n$  can be represented by a set of  $n$  sequences  $S_1, \dots, S_n$  (or strings  $ST_1, \dots, ST_n$ ) following the process detailed in Table 1.

**Table 3**  
 $L_w^H$  and  $NO^H$  per MC signal

	$L_w^H$	$NO^H$		$L_w^H$	$NO^H$
$X_1$	219	11	$X_{11}$	229	3
$X_2$	220	5	$X_{12}$	215	8
$X_3$	225	10	$X_{13}$	223	20
$X_4$	225	5	$X_{14}$	214	12
$X_5$	217	16	$X_{15}$	218	9
$X_6$	240	21	$X_{16}$	225	12
$X_7$	228	6	$X_{17}$	214	7
$X_8$	225	7	$X_{18}$	223	5
$X_9$	221	16	$X_{19}$	229	6
$X_{10}$	210	6	$X_{20}$	231	5

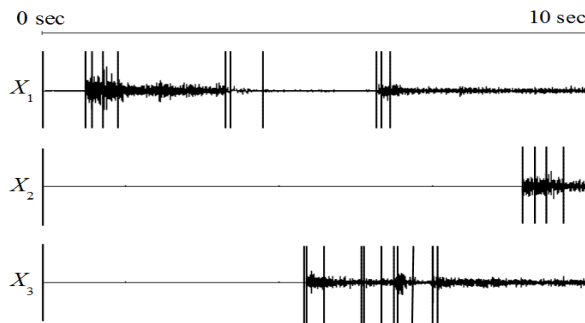


Fig. 8 The events detected for the *EMG* signals  $X_1, X_2, X_3$ .

The goal is to classify the *EMG* signals according to the succession of hand positions. In general the transition between two specific hand positions is not unique and this part of signal is not the objective of our analysis. That is why we need to filter the events related to the transitions from a hand position to other one (Step 7).

To filter these events, it is used Table 4 that summarizes the outcome of the step 6. It represents the *TNI* per group using all the intervals from all the signals. The events of the transitions correspond to the groups with the smaller *TNI* and consequently it is easy to filter them. For each *EMG MC* signal, it is selected the groups of intervals with the highest *TNI*, and that furthermore, the neighboring groups of intervals have a smaller *TNI*. For each hand movement, the *TNI*

increases during the transition until it reaches a hand position. Consequently, in this case, it is not necessary to use a ratio as said in Step 7. After filtering, only the groups of intervals  $G_2$ ,  $G_4$ ,  $G_8$  and  $G_9$  are considered.

**Table 4**  
Total number of intervals per group

	$TNI$		$TNI$
$G_1$	84	$G_6$	191
$G_2$	852	$G_7$	14
$G_3$	165	$G_8$	3195
$G_4$	847	$G_9$	736
$G_5$	9	$G_{10}$	6

Occasionally, events located in the transitions between hand positions could be characterized as a hand position. As it is possible, the filtering process finishes eliminating per  $EMG MC$  signal any event  $E$  with a  $d_E < L_w^H$ . These hand positions are considered as not significant for the classification.

Fig. 9 shows the events (and their characterization) of three  $EMG$  signals after the step 7. And Table 5 shows the new strings obtained.

According to the option 1 in step 8, the pairwise distances between the  $n$  strings  $NST_1, \dots, NST_n$  (using (3) and (4)) are saved in the distance array  $D_{S1}$ . The set of strings  $NST_1, \dots, NST_n$  of Table 5 can be classified automatically as stated in step 9 (Solution 1 or 2).

The second option in step 8 uses  $HMM$  and the sequences  $NS_1, \dots, NS_n$ . Table 6 shows the  $\text{Log}P(NS_i|\lambda_4)$  for each  $NS_i$  with respect to the  $HMM$  model  $\lambda_4$ . The pairwise distances of these values ( $D_{S2}$ ) can be used to classify automatically the  $MC$  signals using the Solution 1 in step 9. In this example, it is not necessary to use other models  $\lambda_{NH}$ .

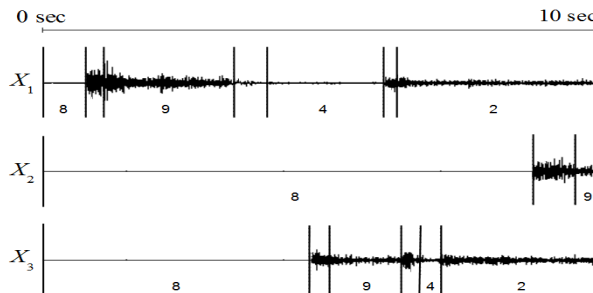


Fig. 9 The events considered for the  $EMG$  signals  $X_1$ ,  $X_2$ ,  $X_3$ . After the step 7.

**Table 5**  
Strings of the *EMG MC* signals

	Strings		Strings
$NST_1$	8942	$NST_{11}$	84
$NST_2$	89	$NST_{12}$	8294
$NST_3$	8942	$NST_{13}$	8942
$NST_4$	84	$NST_{14}$	829
$NST_5$	8924	$NST_{15}$	8294
$NST_6$	8924	$NST_{16}$	8924
$NST_7$	89	$NST_{17}$	849
$NST_8$	84	$NST_{18}$	82
$NST_9$	8942	$NST_{19}$	842
$NST_{10}$	82	$NST_{20}$	8429

**Table 6**  
 $\text{Log } P(NS_i|\lambda_i)$  for the new sequences

	$\text{Log } P(NS_i \lambda_i)$		$\text{Log } P(NS_i \lambda_i)$
$NS_1$	-2.2424	$NS_{11}$	-1.2151
$NS_2$	-0.8437	$NS_{12}$	-3.6332
$NS_3$	-2.2424	$NS_{13}$	-2.2424
$NS_4$	-1.2151	$NS_{14}$	-2.9368
$NS_5$	-2.2696	$NS_{15}$	-3.6332
$NS_6$	-2.2696	$NS_{16}$	-2.2696
$NS_7$	-0.8437	$NS_{17}$	-2.4549
$NS_8$	-1.2151	$NS_{18}$	-1.2976
$NS_9$	-2.2424	$NS_{19}$	-1.8655
$NS_{10}$	-1.2976	$NS_{20}$	-3.6890

*Solution 1 in step 9:* Using  $D_S=D_{S1}$  (or  $D_S=D_{S2}$ ) and the *HC* algorithm, it is possible to calculate different partitions of the strings  $NST_1, \dots, NST_n$  (or sequences  $NS_1, \dots, NS_n$ ) for different group numbers from  $k=2$  to 15. A validity index as Silhouette along with these partitions and  $D_S$ , allows us to determine the optimal number of clusters  $N_O=10$ .

Consequently, the partition for  $k=10$  is the classification of the *EMG MC* signals  $X_1, \dots, X_n$ .

The labels of the classification are shown in Table 7. Table 8 shows the contents of the different clusters.

**Table 7**  
EMG MC signals classification

EMG Signal	$X_1$	$X_2$	$X_3$	$X_4$	$X_5$
Labels	3	10	3	7	4
EMG Signal	$X_6$	$X_7$	$X_8$	$X_9$	$X_{10}$
Labels	4	10	7	3	9
EMG Signal	$X_{11}$	$X_{12}$	$X_{13}$	$X_{14}$	$X_{15}$
Labels	7	2	3	8	2
EMG Signal	$X_{16}$	$X_{17}$	$X_{18}$	$X_{19}$	$X_{20}$
Labels	4	6	9	5	1

**Table 8**  
EMG MC signals classified into 10 clusters

CLUSTERS	EMG MC SIGNALS
$G_1$	$\{X_{20}\}$
$G_2$	$\{X_{12}, X_{15}\}$
$G_3$	$\{X_1, X_3, X_9, X_{13}\}$
$G_4$	$\{X_5, X_6, X_{16}\}$
$G_5$	$\{X_{19}\}$
$G_6$	$\{X_{17}\}$
$G_7$	$\{X_4, X_8, X_{11}\}$
$G_8$	$\{X_{14}\}$
$G_9$	$\{X_{10}, X_{18}\}$
$G_{10}$	$\{X_2, X_7\}$

*Solution 2 in step 9:* It has been selected for the SC a scale parameter  $\sigma=1$ . Fig. 10 shows the correlation  $C_k$  between each  $D_k^U$  and  $D_{S1}$  from  $k=2$  to 15. The value of  $C^H = 0,97$  for  $k = 4$ . So  $U^H = U_4$ .

Again the index Silhouette determines  $N_o=10$  using  $M_{HC}$  and the matrix  $U^H$ .

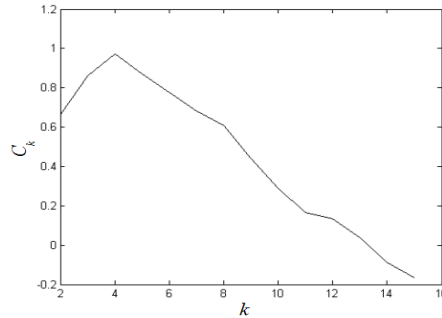


Fig. 10 Correlation  $C_k$  between each  $D_k^U$  and  $D_s$

## 5. Discussion

The proposed algorithm classifies the *EMG MC* signals in a correct way. None of the signals have been misclassified. This classification is defined by using the labels of Table 7.

In step 2, two autoregressive coefficients were used to form the feature vector for each channel  $q$  and window  $W$ . They are good candidates to detect events but that does not prevent to use other kind of coefficients. The proposed method is useful for any selection of the coefficients.

The method makes use of different kind of transformations. We have transformed the  $n$  *EMG MC* signals  $X_1, \dots, X_n$  into a set of  $n$  temporal sequences  $S_1, \dots, S_n$ , then into a set of  $n$  strings  $ST_1, \dots, ST_n$ , and finally (after the filtering process) into a new set of  $n$  strings  $NST_1, \dots, NST_n$ . Using the Solution 2 of step 9, the  $n$  strings are transformed into  $n$  feature vectors of the same size.

From the sequences  $NS_1, \dots, NS_n$ , other set of  $n$  feature vectors (with the same size)  $FS_1, \dots, FS_n$  was obtained by means of the second option in step 8.

As stated in step 8, the distance matrix can be devised from a set of strings or from a set of feature vectors that represent the *MC* signals. In the second option, instead of computing a distance matrix using the feature vectors, it is also possible to use a measure of distances between models (Bicego, Murino, & Figueiredo, 2004).

The *EMG* classification can be used to study different matters related to *EMG* activity but our goal has been to use these signals to validate our method. In this way, we can use our unsupervised method with other kind of *MC* signals where the events nature is unknown.

The processing time and the characteristic parameters ( $CCC$ ,  $N_L$ , ...) from the steps 1 to 4 of *UMEDC* change for each type of *MC* signal because the search of the optimal window size is variable. However this procedure allows to detect events and to classify *MC* signals in an unsupervised way. This means that it can

be used in *MC* signals with unknown events. Supervised methods can not be used because they need to know the nature of the events to develop a training process. Also *UMEDC* does not need to formulate hypotheses regarding the behavior of the events. Consequently *UMEDC* is a generic and alternative way to *HMM* for event detection and signal segmentation into states.

The *UMEDC* method is used for the offline classification of *MC* signals. It is not adapted to be used in real time. However once a classification of a significant set of *MC* signals has been performed, the intervals from all the *MC* signals could train a Support Vector Machine (*SVM*). In this case, the *SVM* could be used to detect events and consequently classify *MC* signals in real time.

The *UMEDC* outcome contains groups of *MC* signals according with their behavior. For each group it is possible to select the sequence of events and analyze some characteristics. For example, the mean and variance of the duration between two specific events, the subsequences that precede a special event (to know if it is unique), if there is an unique event characteristic per group etc.

As it was noticed in the section 2, it is clear that any *CC* over any time interval  $[tm_1, tm_2]$  is not able to classify the *EMG MC* signals according with their behavior and consequently a *CC* partition would be different from the labels of Table 7. To quantify these differences we use the Normalized Mutual Information (*NMI*) (Strehl & Ghosh, 2002). For example, it can be chosen the *CC* partition over the interval [7000 7225] calculated using *HC* for 10 clusters and a *AR(2)* as feature vector per channel. The *NMI* between this *CC* partition and Table 7 is 0.5980. This value is far from the value 1. Other intervals would give us similar results. It shows how a *CC* partition is not able to classify the *EMG* signals correctly. A good result using *CC* over an interval is not impossible but it would be a stroke of luck.

## 6. Conclusion

Classic methods of Classification (*CC*) do not work when *MC* signals have events with start points and durations changing randomly. In these cases, in order to classify correctly, it is necessary to develop methods for events detection and classification.

We have generated a group of *EMG MC* signals where each one is made up of events distributed in a random order.

A new method to detect events has been applied to these *EMG MC* signals. The event detection along with its global characterization allow transforming each *MC* signal with events into a temporal sequence represented by a string. Finally the *MC* signals have been classified automatically in a simple and efficient manner.

These *EMG MC* signals have been used to validate our method. In this way, this method can be used to classify other kind of *MC* signals whose events are unknown a priori.



The classification of the *MC* signals with unknown events by an expert becomes a tedious and difficult task. However, the *UMEDC* allows to detect events and to classify *MC* signals in an unsupervised way. In this way, it is not necessary to know the nature of the events and formulate hypotheses regarding their behavior.

The *UMEDC* is an offline method and can not be applied to detect events and classify *MC* signals in real time. However the intervals from a significant set of *MC* signals classified using *UMEDC* could be used to train a *SVM*. This *SVM* could classify new events and *MC* signals in real time.

Depending of the kind of *MC* signal, the time processing of the *UMEDC* for event detection could be significant. New strategies to determine the grid search for the window length and the optimal window size are welcome.

The *UMEDC* can be applied in various domains such as Bioengineering, Volcanology, Geophysics, Nuclear Fusion etc. This method has the ability to improve human thought and reasoning, specially for non-stationary random *MC* signals. It turns the information of these *MC* signals into sequences of easy interpretation.

The event detection part of *UMEDC* can be used to capture frequency and duration of events. For example to analyze *EEG* artifacts for psychophysiological studies. When the *MC* signals contains some extreme and/or special events, *UMEDC* allows to identify the sequence of characteristic events that precede them. Each sequence can be used for prediction and to decide either following the sequence or avoiding it. For example, a disruption in Nuclear Fusion is a special event to avoid. The *UMEDC* outcome could also be used for anomaly detection (or outlier detection). It means detecting events or *MC* signals which do not conform to a significant *UMEDC* outcome.

## Acknowledgements

This work was supported in part by the Spanish Ministry of Economy and Competitiveness under Project ENE2012-38970-C04-01/03, the Project DPI2011-27818-C02-02 and FEDER funds.

## References

- Aggarwal, C. C. (2002). On effective classification of strings with wavelets. *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 163–172.
- Antunes, C. M., & Oliveira, A. L. (2001). Temporal data mining: An overview. *KDD Workshop on Temporal Data Mining*, 1-13.
- Bartlett, P. L., Ben-David, S., & Kulkarni, S. R. (2000). Learning changing concepts by exploiting the structure of change. *Machine Learning*, 41(2), 153–174. <http://doi.org/10.1023/A:1007604202679>

- Bicego, M., Murino, V., & Figueiredo, M. a T. (2004). Similarity-based classification of sequences using hidden Markov models. *Pattern Recognition*, 37(12), 2281–2291. <http://doi.org/10.1016/j.patcog.2004.04.005>
- Bifet, A., Gavaldà, R., & Gavaldà, R. (2007). Learning from Time-Changing Data with Adaptive Windowing. *SDM*, 7, 2007. <http://doi.org/10.1137/1.9781611972771.42>
- Burg, J. P. (1975). Maximum Entropy Spectral Analysis. In *Proceedings of the 37th Annual International SEG Meeting* (Vol. 6, p. 0).
- Chan. (n.d.). Myoelectric Control Development Toolbox. Retrieved from [www.sce.carleton.ca/faculty/chan](http://www.sce.carleton.ca/faculty/chan)
- Chan ADC, G. G. (2007). Myoelectric control development toolbox. In *30th Conference of the Canadian Medical & Biological Engineering Society*. Toronto.
- Cheng, B. Y. M., Carbonell, J. G., & Klein-Seetharaman, J. (2005). Protein classification based on text document classification techniques. *Proteins*, 58(4), 955–970. <http://doi.org/10.1002/prot.20373>
- Cormode, G., & Muthukrishnan, S. (2004). What's new: finding significant differences in network data streams. *IEEE INFOCOM 2004*, 3, 1219–1232. <http://doi.org/10.1109/TNET.2005.860096>
- Dias, J. G., & Ramos, S. B. (2014). Dynamic clustering of energy markets: An extended hidden Markov approach. *Expert Systems with Applications*. <http://doi.org/10.1016/j.eswa.2014.05.030>
- Gama, J., Medas, P., Castillo, G., & Rodrigues, P. (2004). Learning with drift detection. *Advances in Artificial Intelligence–SBIA 2004*, 286–295. [http://doi.org/10.1007/978-3-540-28645-5\\_29](http://doi.org/10.1007/978-3-540-28645-5_29)
- Gehrke, J., Ganti, V., Gehrke, J., & Ramakrishnan, R. (1999). A Framework for Measuring Changes in Data Characteristics. *Focus*, 578, 126–137. <http://doi.org/10.1145/303976.303989>
- Halkidi, M., & Vazirgiannis, M. (2001). Clustering Validity Assessment: Finding the optimal partitioning of a data set. In *In Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on* (pp. 187–194). <http://doi.org/10.1109/ICDM.2001.989517>
- Ho, S. (2005). A martingale framework for concept change detection in time-varying data streams. *Proceedings of the 22nd International Conference on Machine Learning (ICML-05)*, (2004), 321–327. <http://doi.org/10.1145/1102351.1102392>
- J. J. R. Diez, C. A. González, and H. B. (2001). Boosting interval based literals. *Intell. Data Anal*, 5(3), 245–262.
- Ji, X., Bailey, T. L., & Dong, G. (2007). Mining minimal distinguishing subsequence patterns with gap constraints. *Knowledge and Information Systems*, 11, 259–286. <http://doi.org/10.1109/ICDM.2005.96>

- K. Santhisree1, A. Damodaram, S. V. A. (2010). An Enhanced DbSCAN Algorithm to Cluster Web usage Data using Rough Sets and Upper Approximations. *International Journal of Computer Science & Communication*, 1(1), 263–265.
- Kaján, L., Kertész-Farkas, A., Franklin, D., Ivanova, N., Kocsor, A., & Pongor, S. (2006). Application of a simple likelihood ratio approximant to protein sequence classification. *Bioinformatics*, 22(23), 2865–2869. <http://doi.org/10.1093/bioinformatics/btl512>
- Kifer, D., Ben-david, S., & Gehrke, J. (2004). Detecting Change in Data Streams. In *Proceedings of the 30th VLDB Conference* (pp. 180–191).
- Krishnamurthy, B., Sen, S., Zhang, Y., & Chen, Y. (2003). Sketch-based change detection: methods, evaluation, and applications. *Proceedings of the 3rd ACM SIGCOMM Conference on Internet Measurement*, 234–247. <http://doi.org/10.1145/948205.948236>
- Lawhern, V., Hairston, W. D., & Robbins, K. (2013). DETECT: a MATLAB toolbox for event detection and identification in time series, with applications to artifact detection in EEG signals. *PloS One*, 8(4), e62944. <http://doi.org/10.1371/journal.pone.0062944>
- Li, Z., Fang, H., & Huang, M. (2015). Diversified learning for continuous hidden Markov models with application to fault diagnosis. *Expert Systems with Applications*, 42(23), 9165–9173. <http://doi.org/10.1016/j.eswa.2015.08.027>
- Littlestone, N. (1988). Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2(4), 285–318. <http://doi.org/10.1007/BF00116827>
- Mitsa, T. (2010). *Temporal Data Mining. Clinics in Laboratory Medicine* (Vol. 28).
- Mohd. Shahnawaz, Ashish Ranjan, M. D. (2011). Temporal Data Mining: An Overview. *International Journal of Engineering and Advanced Technology (IJEAT)*, 1(1), 2249 – 8958.
- Needleman, S. B., & Wunsch, C. D. (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3), 443–453. [http://doi.org/10.1016/0022-2836\(70\)90057-4](http://doi.org/10.1016/0022-2836(70)90057-4)
- Ng, A. Y., Jordan, M. I., & Weiss, Y. (2001). On Spectral Clustering: Analysis and an algorithm. *Advances in Neural Information Processing Systems*, 849–856. <http://doi.org/10.1.1.19.8100>
- P. Kumar, P. Radha Krishna, B. S. R. (2010). A New Similarity Metric for Sequential Data. *A New Similarity Metric for Sequential Data. International Journal of Data Warehousing and Mining*, 6(4), 16–32.
- R. Durbin, S. R. Eddy, A. Krogh, and G. M. (1998). No Title. In *Probabilistic Models of Proteins and Nucleic Acids* (pp. 47–65).
- Rabiner, L. R. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*. <http://doi.org/10.1109/5.18626>

- Rokach, L., & Maimon, O. (2005). Clustering Methods Data Mining and Knowledge Discovery Handbook. *Data Mining and Knowledge Discovery Handbook*, 321–352. <http://doi.org/10.1007/978-0-387-09823-4>
- Rousseeuw, P. J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20, 53–65. [http://doi.org/10.1016/0377-0427\(87\)90125-7](http://doi.org/10.1016/0377-0427(87)90125-7)
- S. Muthukrishnan, Eric van den Berg, Y. W. (2007). Sequential Change Detection on Data Streams. In *Seventh IEEE International Conference on Data Mining Workshops* (pp. 541–551).
- Sokal, R. R., & Rohlf, F. J. (1962). The Comparison of Dendrograms by Objective Methods. *Taxon*, 11(2), 33–40. <http://doi.org/10.2307/1217208>
- Srivastava, P. K., Desai, D. K., Nandi, S., & Lynn, A. M. (2007). HMM-ModE--improved classification using profile hidden Markov models by optimising the discrimination threshold and modifying emission probabilities with negative training sequences. *BMC Bioinformatics*, 8, 104. <http://doi.org/10.1186/1471-2105-8-104>
- Strehl, A., & Ghosh, J. (2002). Cluster Ensembles – A Knowledge Reuse Framework for Combining Multiple Partitions. *Journal of Machine Learning Research*, 3, 583–617. <http://doi.org/10.1162/153244303321897735>
- V. Lawhern, WD. Hairston, K. McDowell, M. Westerfield, K. R. (2012). Detection and classification of subject-generated artifacts in EEG signals using autoregressive models. *Neuroscience Methods*, 181–189. <http://doi.org/10.1016/j.jneumeth.2012.05.017>
- Wald, A. (2004). *Sequential Analysis*, Dover Publications.
- Weston, J., Leslie, C., Le, E., Zhou, D., Elisseeff, A., & Noble, W. S. (2005). Semi-supervised protein classification using cluster kernels. *Bioinformatics*, 21(15), 3241–3247. <http://doi.org/10.1093/bioinformatics/bti497>
- Widmer, G., & Kubat, M. (1996). Learning in the presence of concept drift and hidden contexts. *Machine Learning*, 23(1), 69–101. <http://doi.org/10.1007/BF00116900>
- Xi, X., Keogh, E., Shelton, C., Wei, L., & Ratanamahatana, C. A. (2006). Fast time series classification using numerosity reduction. In *Proceedings of the 23rd international conference on Machine learning (ICML)* (pp. 1033–1040). <http://doi.org/10.1145/1143844.1143974>
- Xing, Z., Pei, J., & Keogh, E. (2010). A brief survey on sequence classification. *ACM SIGKDD Explorations Newsletter*, 12(1), 40. <http://doi.org/10.1145/1882471.1882478>
- Y Leela Sandhya Rani, P Naga Deepthi, C. R. D. (2013). Clustering Algorithm for Temporal Data Mining: An Overview. *International Journal of Emerging Technology and Advanced Engineering*, 3(7), 2250–2459.
- Yakhnenko, O., Silvescu, A., & Honavar, V. (2005). Discriminatively trained markov model for sequence classification. *Proceedings - IEEE International Conference on Data Mining, ICDM*, 498–505. <http://doi.org/10.1109/ICDM.2005.52>

## *Capítulo 5: ART.2*

- Ye, L., & Keogh, E. (2009). Time series shapelets: a new primitive for data mining. *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 947–956. <http://doi.org/10.1145/1557019.1557122>
- Z. Xing, J. Pei, G. Dong, and P. S. Y. (2008). Mining sequence classifiers for early prediction. *Proceedings of the 2008 SIAM International Conference on Data Mining*, 644–655.




## CAPÍTULO 6

### ART.3: DETERMINATION OF THE OPTIMAL NUMBER OF CLUSTERS USING A SPECTRAL CLUSTERING OPTIMIZATION

Resumen:

- *Revista:* Expert Systems with Applications
- *Current Impact Factor:* 2.981; *5-Year Impact Factor:* 2.879
- *Selección de la primera página del artículo publicado:*


Expert Systems With Applications 65 (2016) 304–314



Contents lists available at [ScienceDirect](#)

### Expert Systems With Applications

journal homepage: [www.elsevier.com/locate/eswa](http://www.elsevier.com/locate/eswa)




---

## Determination of the optimal number of clusters using a spectral clustering optimization

Angel Mur<sup>a,\*</sup>, Raquel Dormido<sup>a</sup>, Natividad Duro<sup>a</sup>, Sebastian Dormido-Canto<sup>a</sup>, Jesús Vega<sup>b</sup>

<sup>a</sup>Department of Computer Sciences and Automatic Control, UNED, Juan del Rosal 16 - 28040 Madrid, Spain  
<sup>b</sup>National Fusion Laboratory by Magnetic Confinement, CIEMAT, Complutense 40 - 28040 Madrid, Spain



---

**ARTICLE INFO**

*Article history:*  
Received 18 March 2016  
Revised 23 July 2016  
Accepted 20 August 2016  
Available online 26 August 2016

*Keywords:*  
Spectral clustering  
Optimal number of clusters  
Silhouette Index  
Local scaling  
Image segmentation

**ABSTRACT**

In this paper, we present a new method, called Spectral Global Silhouette method (GS), to calculate the optimal number of clusters in a dataset using a Spectral Clustering algorithm. It combines both a Silhouette Validity Index and the concept of Local Scaling. First, the GS algorithm has first been tested using synthetic data. Then, it is applied on real data for image segmentation task. In addition, three new methods for image segmentation and two new ways to calculate the optimal number of groups in an image are proposed. Our experiments have shown a promising performance of the proposed algorithms.

© 2016 Elsevier Ltd. All rights reserved.

## Determination of the optimal number of clusters using a spectral clustering optimization

Angel Mur<sup>a</sup>, Raquel Dormido<sup>a</sup>, Natividad Duro<sup>a</sup>, Sebastian Dormido-Canto<sup>a</sup>, Jesús Vega<sup>b</sup>

<sup>a</sup> Department of Computer Sciences and Automatic Control, UNED, Juan del Rosal 16 - 28040 Madrid, Spain, [a.r.m.g@outlook.fr](mailto:a.r.m.g@outlook.fr), [raquel@dia.uned.es](mailto:raquel@dia.uned.es), [nduro@dia.uned.es](mailto:nduro@dia.uned.es), [sebas@dia.uned.es](mailto:sebas@dia.uned.es)

<sup>b</sup> National Fusion Laboratory by Magnetic Confinement. CIEMAT, Complutense 40 - 28040 Madrid, Spain, [jesus.vega@ciemat.es](mailto:jesus.vega@ciemat.es)

### ABSTRACT

In this paper, we present a new method, called Spectral Global Silhouette method (*GS*), to calculate the optimal number of clusters in a dataset using a Spectral Clustering algorithm. It combines both a Silhouette Validity Index and the concept of Local Scaling. First, the *GS* algorithm has first been tested using synthetic data. Then, it is applied on real data for image segmentation task. In addition, three new methods for image segmentation and two new ways to calculate the optimal number of groups in an image are proposed. Our experiments have shown a promising performance of the proposed algorithms.

**Keywords:** Spectral Clustering, Optimal Number of Clusters, Silhouette Index, Local Scaling, Image Segmentation

### 1. Introduction

Clustering is an unsupervised learning method that divides data into groups (clusters) that are meaningful and/or useful. When the objective is to find meaningful groups, then the clusters should capture the natural structure of the data (Everitt, Landau, Leese, & Stahl, 2011).

Cluster analysis plays an important role in a wide variety of fields (e.g. see (Jain, Murty, & Flynn, 1999; Xu & Wunsch, 2005)): social sciences, biology, statistics, pattern recognition, information retrieval, machine learning and data mining.

Several clustering methods with different characteristics have been proposed for different purposes. Some well-known clustering algorithms are: *K*means (MacQueen, 1967), *EM* (Expectation Maximization) (Dempster, Laird, & Rubin, 1977), Hierarchical clustering algorithms (*HC*) (Rokach & Maimon, 2005) and Spectral clustering (*SC*) (Luxburg, 2007; Ng, Jordan, & Weiss, 2001). In all of them, the estimation of the number of clusters contained in a dataset is an essential issue. The user has to define the number of clusters either a priori or a posteriori.

In practical problems, the number of clusters is generally unknown. A simple approach to find the optimal number consists of getting a set of data partitions with different numbers of clusters and then to select the partition that provides the



best result according to a specific validity index (*VID*). With the help of this *VID* the optimal number of clusters is automatically determined.

Some of the most well-known *VID* are the Davies-Bouldin Index (Davies & Bouldin, 1979), the Calinski-Harabasz Index (Calinski & Harabasz, 1974), Dunn's Index (Dunn, 1974), the Silhouette Index (Rousseeuw, 1987), the *S\_Dbw* Validity Index (Halkidi & Vazirgiannis, 2001) etc.

Spectral clustering (*SC*) is one of the most popular clustering methods. This method can be applied by using standard linear algebra techniques and it usually provides meaningful groups. It should be noted that, typically, the number of clusters is set in a manual way. However, approaches to automatically determine the optimal number of clusters are always preferred. This is one of the main objectives of the present article: determining the optimal number of clusters using a *SC* algorithm.

Ref. (Zelnik-manor & Perona, 2004) proposes a spectral clustering algorithm that computes automatically the optimal number of groups. It can also handle multi-scale data using the concept of local scaling. To determine the optimal number of groups, this algorithm minimizes the cost of aligning a set of eigenvectors with a canonical coordinate system (using rotations). Ref. (Xiang & Gong, 2008) proposes an alternative method to estimate the number of clusters. To this end, the more significant eigenvectors to get separated data (using *EM* algorithm) are selected. Both methods are applied to image segmentation.

Image segmentation is the process of assigning a label to each pixel in an image in such a way that pixels with the same label share certain characteristics (ex: colour, intensity, or texture). The goal of segmentation is to simplify the representation of an image into something that is meaningful and easier to analyse.

The use of Spectral clustering for image segmentation, is computationally intensive. This is due to the use of an affinity matrix (*A*) that contains all the pairwise affinities between pixels. References (Fowlkes, Belongie, Chung, & Malik, 2004; Shi & Malik, 1998; Tung, Wong, & Clausi, 2010) provide different approaches to reduce the computational requirements. For instance, (Shi & Malik, 1998) uses a sparse version of *A* in which each element is connected only to a few of its nearby neighbours in the image and all other connections are assumed to be zero. A different approach is used in (Fowlkes et al., 2004): the pairwise similarities from a small random subset of pixels are used. Last but not least, (Tung et al., 2010) combines a blockwise segmentation strategy along with a stochastic ensemble consensus.

In the present work, a simple and effective method named Spectral Global Silhouette method is shown. *GS* uses *SC* together with the Silhouette Validity index and the concept of local scaling. This combination allows finding the optimal number of clusters in a data set as well as an optimal local scaling.

The application of the *SC* algorithm to a set of data points provides new representations of these data points with the help of the largest eigenvectors of a

data affinity matrix. The present article works directly on these new representations of data points unlike (Zelnik-manor & Perona, 2004) and (Xiang & Gong, 2008) that initially work with the eigenvectors either analysing their structure or selecting the more relevant.

For large datasets, the *GS* algorithm requires high computational resources. In this article, the image segmentation problem to illustrate how to apply *GS* for a large dataset is presented. Two methods are shown: *WA* and *WB*. Both of them are based on *GS* and are used to calculate the optimal number of groups in an image (*IM*). The *WA* algorithm uses a reduced version of *IM*. The *WB* algorithm uses a scalable approach. The proposed methods *WA* and *WB* are then applied to the image segmentation problem resulting in different algorithms. *WA* is applied in combination with an optimal sparse version of *A* (Shi & Malik, 1998) (*GSWA* method). Other simple method for image segmentation (*GSWB*) that uses *WB* is also presented. *GSWB* is compared and validated by means of the Nyström method (Fowlkes et al., 2004). The *WB* algorithm together with the Nyström method form the Nyström\_ *WB* method that is also analysed.

In Section 2 some concepts used in the paper are reviewed. In section 3, the proposed methods *GS*, *WA*, *WB*, *GSWA*, *GSWB* and Nyström\_ *WB* are described. In section 4 these methods are tested and validated using synthetic and real data. Finally in sections 5 and 6, a discussion and conclusions of the paper are respectively presented.

## 2. Background

This section reviews some well-known methods used in the paper: Hierarchical clustering (*HC*) (Rokach & Maimon, 2005), *K*means clustering (MacQueen, 1967), Spectral Clustering (*SC*), Local Scaling transformation (Zelnik-manor & Perona, 2004) and the Silhouette Validity index (Rousseeuw, 1987) (Average Silhouette index (*AS*) or Simplified Silhouette index (*SS*) (Hruschka & Covões, 2005).

### 2.1 Hierarchical and *K*means Clustering

Hierarchical clustering (*HC*) groups data over a variety of scales by creating a cluster tree or dendrogram. It follows several steps: 1) find the similarity or dissimilarity between every pair of objects in the data set 2) grouping the objects into a binary, hierarchical cluster tree (linkage) 3) determining where to cut the hierarchical tree into clusters.

In this paper, the *HC* computes the distance between two data points and the distance between two clusters (for the linkage) using respectively the Euclidean and the Average distance (this means the average of the distances of each element of the cluster with each element of the other cluster). It is important to note that this is the only choice made for the *HC* algorithm.

Kmeans clustering is a partitioning method. By dividing data into  $k$  sub-clusters, Kmeans represents all the data by the mean values or centroids of their respective sub-clusters. The selection of the initial centres in each sub-cluster is randomly chosen or derived from some heuristic. The algorithm follows an iterative process where each iteration associates every data point to its nearest centroid. This is carried out according to some chosen distance metric. The new centroids are calculated by taking the mean of all the data points within each sub-cluster. The algorithm iterates until no data points move from one sub-cluster to another.

Unlike hierarchical clustering, Kmeans clustering operates on actual observations (rather than the larger set of dissimilarity measures), and creates a single level of clusters. This distinction means that Kmeans clustering is often more suitable than hierarchical clustering for large amounts of data.

The algorithms *HC* and *Kmeans* are used in the last step of the *SC* algorithm.

## 2.2 Spectral Clustering

The goal of Spectral Clustering is to cluster a set of data points  $x_1, \dots, x_n$  as a graph partitioning problem without making any assumption on the form of the data clusters. Spectral clustering often produces better results than classical clustering algorithms such as Kmeans and mixture models. It also allows finding non-convex clusters.

Different stages are involved in the Spectral Clustering algorithm. 1) A pre-processing step to construct the graph and the affinity matrix representing the data set. 2) The calculation of the spectral representation. To this end, it forms the associated Laplacian matrix and computes its eigenvalues and eigenvectors. Then, it maps each data point to a lower-dimensional representation based on two or more eigenvectors. 3) The clustering process that assign points to two or more classes, based on the new representation.

So, given a set of points  $x_1, \dots, x_n$  to be partitioned into  $k$  clusters  $G_1, \dots, G_k$  the spectral clustering can be formulated as follows (Luxburg, 2007):

1. Calculate the affinity matrix  $A$  defined by (1)

$$A(i, j) = e^{-\frac{d^2(x_i, x_j)}{2\sigma^2}} \quad \text{for } i \neq j$$

$$\text{and } A(i, i) = 0 \quad (1)$$

where  $d(x_i, x_j)$  is the distance between  $x_i$  and  $x_j$  and  $\sigma$  is a scaling parameter.

2. Construct the normalized Laplacian matrix  $L_{\text{sym}} = Q^{-1/2} A Q^{-1/2}$ , where  $Q$  is a diagonal matrix whose  $(i, i)$ -element is the sum of  $A$ 's  $i$ -th row.

3. Find the  $k$  largest eigenvectors of  $L_{sym}$  (eigenvectors whose eigenvalues are the largest in magnitude) and form the matrix  $U_k$  by stacking the eigenvectors in columns:  $U_k = [u_1 \dots u_k] \in R^{n \times k}$ .
4. Form the matrix  $Y_k$  from  $U_k$  by normalizing the rows of  $U_k$  to have unit length.
5. Treat each row of  $Y_k$  as a point in  $R^k$  and cluster them into  $k$  groups  $C_1, \dots, C_k$  via  $K$ means (or  $HC$ ).
6. Assign the original points  $x_i$  to cluster  $G_j$  if and only if row  $i$  of the matrix  $Y_k$  was assigned to cluster  $C_j$ .

### 2.3 Local Scaling

Spectral clustering requires the construction of a weighted graph that encodes the similarity (or affinity) between data points. Sometimes the estimation of affinities fails when trying to capture the data structure. To solve this problem a local scaling is proposed in (Zelnik-manor & Perona, 2004).

As equation (1) shows, the affinity matrix  $A$  of a given set  $S = \{x_1, \dots, x_n\}$  of data points depends on the scaling parameter  $\sigma$ . This scaling parameter  $\sigma$  is some measure of when two data points are considered similar. The selection of  $\sigma$  has a high impact on the clustering. As ref. (Zelnik-manor & Perona, 2004) proposes, instead of selecting a single scaling parameter  $\sigma$ , a local scaling parameter  $\sigma_i$  for each data point  $x_i$  can be calculated. In this context, the affinity between a pair of data points is rewritten as

$$A(i, j) = e^{\frac{-d^2(x_i, x_j)}{\sigma_i \sigma_j}} \quad \text{for } i \neq j$$

$$\text{and } A(i, i) = 0 \quad (2)$$

A choice of  $\sigma_i$  is

$$\sigma_i = d(x_i, x_N) \quad (3)$$

where  $x_N$  is the  $N$ 'th neighbour of data point  $x_i$ . For example, for the point  $x_i$  the distances between  $x_i$  and the other points are calculated. Then, these distances are sorted and the distance of the  $N$ 'th position is selected. This distance is  $\sigma_i$ .

The value of  $N$  used by (Zelnik-manor & Perona, 2004) for any set of data is  $N = 7$ . This value does not necessarily have to be unique. The search of an optimal value for  $N$  can improve the classification of the data.

Local scaling is especially useful when the classification of the data contains multiple scales (for example if a cluster is tight, other is sparse...). By means of local scaling, the affinities across clusters are significantly lower than the

affinities within any single cluster. Furthermore, local scaling improves the behaviour of the Silhouette index (described below) which is used for cluster validation.

#### 2.4 Silhouette Index

Cluster validation allows assessing the quality of clustering results. These clustering results are usually measured in terms of cohesion and separation of clusters given by some indices. Cohesion determines how closely the data are in a cluster. Separation indicates how well-separated a cluster is from other clusters. A well-known index to validate a clustering partition is the Silhouette index. It is based on geometrical considerations and combines ideas of both cohesion and separation of the clusters.

Let us consider that the  $j$ th data point of a dataset  $S = \{x_1, \dots, x_n\}$ ,  $x_j$ , belongs to a given cluster  $p \in \{1, \dots, k\}$ . The procedure for calculating its silhouette coefficient is as follows:

1. For  $x_j$ , calculate its average distance  $a_j^p$  to all other data points in its cluster  $p$ .
2. For  $x_j$  and any cluster  $q$  not containing the data point  $x_j$  ( $q=1, \dots, k, q \neq p$ ), calculate the data point's average distance  $d_j^q$  to all the data points in the given cluster  $q$ . The minimum with respect to all clusters is  $b_j^p$ .
3. For  $x_j$ , the silhouette coefficient is then given by

$$SH_{x_j} = \frac{b_j^p - a_j^p}{\max\{a_j^p, b_j^p\}} \quad (4)$$

where the denominator is just a normalization term. The value of the silhouette coefficient  $SH_{x_j}$  can vary between -1 and 1. Clearly, the higher  $SH_{x_j}$ , the better the assignment of  $x_j$  to cluster  $p$  is. In case  $p$  is a singleton, i.e., if it is constituted uniquely by  $x_j$ , then it is assumed by convention that  $SH_{x_j} = 0$ .

The average silhouette coefficient of a cluster of  $nc$  data points (for  $j=1 \dots nc$ ) is the average of the silhouette coefficients of data points belonging to the cluster:

$$SHM_p = \frac{1}{nc} \sum_{j=1}^{nc} SH_{x_j} \quad (5)$$

The average silhouette coefficient of all data points ( $AS$ ) is an overall measure of the goodness of a clustering: the higher value, the better.

This index works well with  $HC$  and  $K$ means clustering. It is used to determine the optimal number of clusters. The  $AS$  is also robust for noisy data.

The  $AS$  depends on the computation of all distances among all data points. Its computational cost is  $O(en^2)$  where  $e$  is the number of features that define the data points and  $n$  is the number of data points. For a large value of  $n$ , it is better to replace the  $AS$  with the Simplified Silhouette index ( $SS$ ) (Hruschka & Covões, 2005). The  $SS$  is based on distances among data points and cluster centroids. To calculate  $SS$  the equation (4) is redefined: the distance between the data point  $x_j$  and the centroid of its cluster  $p$  is  $a^p_j$ . The distance between the data point  $x_j$  and the centroid of the cluster  $q$  (not containing the data point  $x_j$ ) is  $d^q_j$ . The minimum  $d^q_j$  with respect to all clusters is  $b^p_j$ .

The computational cost of  $SS$  is  $O(kne)$  where  $k$  is the number of clusters. Normally,  $k$  is significantly less than  $n$ .

### 3. Proposed Algorithms

#### 3.1 Spectral Global Silhouette Method

In this section, the proposed Spectral Global Silhouette Method ( $GS$ ) is presented. This method addresses the problem of automatically finding the optimal number of clusters ( $NO$ ) for the spectral clustering. It makes use of the Silhouette Validity index ( $AS$  or  $SS$ ) and the concept of local scaling. Together with  $NO$ , its corresponding optimal  $N^{\text{th}}$  neighbour ( $ON$ ) of the local scaling is determined.

Given a set of data points and a set of neighbour values of the local scaling, the spectral clustering requires the construction of new representations that encodes the similarity between the data points. The  $GS$  searches for the representation that provides the best compactness and separation of the clusters. The highest Silhouette coefficient determines the best representation together with  $NO$  and  $ON$ .

The proposed algorithm makes use of the following elements:

- 1) A set of data points  $S = \{x_1, \dots, x_n\}$ .
- 2) Related to the local scaling (see section 2.3):
  - A set of  $M$  values for  $N$ :  $N_m = N_0 + m \times D$  for  $m=0 \dots M-1$  and an integer  $D \geq 5$ .
  - The local scale  $\sigma_i$  for each data point  $x_j \in S$ : it is calculated using (3) and  $N$ .
- 3) Related to the spectral clustering (see section 2.2):
  - An affinity matrix  $A$  (using (2)) and its normalized Laplacian  $L_{sym}$ , for each  $N$  value.
  - The number  $C$ : it represents the largest group number selected.
  - The matrices  $U_k^N$  for each  $L_{sym}$  with  $k=2 \dots C$ :  $U_2^N = [X_{C-1}, X_C] \dots U_C^N = [X_1, \dots, X_C]$ .

- 4) Related to the optimal number of clusters for a particular  $N$  value:
- The array  $KV=[2,3,\dots,C]$ : it contains the  $k$  values used.
  - The maximum Silhouette value  $MSI_k^N$ : For each  $U_k^N$ , it is selected the maximum Silhouette value between the silhouette values of  $C-1$  unsupervised classifications (from 2 to  $C$  groups) of the  $U_k^N$  rows.  $HC$  or  $K$ means can be used in the unsupervised classifications.
  - The optimal number of clusters  $NO_k^N$ : For each  $U_k^N$ ,  $NO_k^N$  corresponds with the number of groups related to the  $MSI_k^N$  value.
  - The array  $VMSI^N$ : it contains the  $MSI_k^N$  values of the  $C-1$  matrices.
  - The array  $VNO^N$ : it contains the  $NO_k^N$  values of the  $C-1$  matrices.
  - The value  $MS^N$ : it is the maximum Silhouette value between the values of  $VMSI^N$  for which their respective  $VNO^N$  matches their  $KV$ .
  - The optimal number of clusters  $NO^N$ : Once it has been selected the position  $T$  of  $MS^N$  in  $VMSI^N$ , the optimal number of clusters  $NO^N=VNO^N(T)$ .
  - The matrix  $OU^N$ :  $OU^N=U_{T+1}^N$ . Its rows represent the set of data points  $S$ .
- 5) Related to the global optimal number of clusters for all the  $N$  values:
- The array  $GVMSI$ : it contains the values  $MS^N$ .
  - The array  $GVNO$ : it contains the values  $NO^N$ .
  - The value  $GMS$ : it is the maximum Silhouette value between the values of  $GVMSI$ .
  - The global optimal number of clusters  $NO$ : it is the value in  $GVNO$  that corresponds with  $GMS$  in  $GVMSI$ .
  - The matrix  $OU$ : it is the matrix related to  $NO$ .

The proposed algorithm is described as follows.

**Algorithm GS. Spectral Global Silhouette Method**

**Input:** A set of data points  $S$ , a value for  $C$ , the array  $KV=[2,3,\dots,C]$  and  $M$  values for  $N$ .

**Output:** Optimal number of clusters  $NO$  and classification of  $S$  into  $NO$  clusters.

**for each  $N$  repeat**

1. Calculate the local scale  $\sigma_i$  for each data point  $x_j \in S$ .
2. Calculate the affinity matrix  $A$  and its normalized Laplacian  $L_{sym}$ .
3. Calculate the matrices  $U_k^N$  for each  $L_{sym}$  with  $k=2,\dots,C$ 
  - for each  $U_k^N$**
  - 4. Calculate the maximum Silhouette value  $MSI_k^N$ .
  - 5. Calculate the optimal number of clusters  $NO_k^N$ .

6. Save the  $MSI_k^N$  value in the  $k$  position of the array  $VMSI^N$ .
  7. Save the  $NO_k^N$  value in the  $k$  position of the array  $VNO^N$ .
- end for**
8. Calculate the  $MS^N$  value, the optimal number of clusters  $NO^N$  and the matrix  $OU^N$ .
  9. Save the  $MS^N$  value in the  $GVMSI$  array and the  $NO^N$  value in the  $GVNO$  array.
- until for a particular  $m=p$ ,  $MS^{Np} > MS^{Np+1}$**
10. Select the optimal neighbour  $ON=N_p$ , the optimal number of clusters  $NO=NO^{Np}$  and the best matrix  $OU=OU^{Np}$ .
  11. Classify the data points  $S$  into  $NO$  clusters using  $OU$  and  $NO$  along with an unsupervised clustering method.

**Explanation:**

First, a value for  $C$  and  $M$  values for  $N$  ( $N_m = N_0 + m \times D$  for  $m=0 \dots M-1$  and the integer  $D \geq 5$ ) are selected.

Second, for each  $N$ , it is calculated its  $MS^N$ ,  $NO^N$  and  $OU^N$ . The values  $MS^N$  and  $NO^N$  are saved in the arrays  $GVMSI$  and  $GVNO$  respectively. This process stops when, for a particular  $m=p$ ,  $MS^{Np} > MS^{Np+1}$ . Finally, the optimal neighbour  $ON=N_p$ , the optimal number of clusters  $NO=NO^{Np}$  and the best matrix  $OU=OU^{Np}$  are selected.

The data points  $S$  can be classified into  $NO$  clusters using  $OU$  and  $NO$  along with any unsupervised clustering method (such as Kmeans or Hierarchical Clustering).

**Some remarks about the algorithm:**

1) The data points  $S$  are centralized and scaled over the maximum absolute value. 2) Normally, the chosen value for  $N_0$  is 5. 3) The normalization of the matrices  $U_k^N$  is not used. 4) It is understood that  $M > p$  and consequently, it is only necessary to calculate  $MS^N$ ,  $NO^N$  and  $OU^N$  for  $m \leq p$ . The optimal  $NO$  selected in  $GVNO$  is also the most recurrent number in this array. 5) The rows of each matrix  $U_k^N$  represent the data points  $S$ . The spectral clustering maps each data point to a  $k$ -dimensional representation using  $k$  eigenvectors. The highest Silhouette value determines which  $U_k^N$  representation has the best compactness and separation of the clusters. Consequently, this value allows finding the optimal number of clusters  $NO$ , the optimal value  $ON$  and its optimal matrix  $OU$ . 6) The  $SS$  index has a good behaviour with the help of the local scaling that improves the separation between clusters.

**3.2 Global Silhouette Method for Image Segmentation**

Image segmentation plays a fundamental role in computer vision area. Several image segmentation algorithms based on spectral clustering have been developed.



One of the challenges of these algorithms is the scalability. When images are sufficiently large spectral clustering becomes computationally intractable. In the same way, the direct application of the proposed *GS* algorithm for image segmentation, is computationally intensive and requires a huge amount of memory.

In this subsection, the application of the *GS* to the image segmentation problem is shown. This is a significant example on how to apply *GS* for a large dataset. To this end, two approaches (*WA* and *WB*) to calculate the *NO* in an image using *GS* are proposed. Then, depending on the approach selected to calculate *NO* different segmentations methods are presented.

### 3.2.1 Methods to calculate the optimal number of clusters (*NO*) in an image

Let *IM* be the image of which we want to determine the optimal number of clusters. The proposed algorithm *WA* applies *GS* to a reduced image of *IM* while the algorithm *WB* uses a set of non-overlapping blocks of *IM*. The two proposed methods are described as follows:

1) **Method *WA*:** First the image *IM* of size ( $S_1 \times S_2$ ) is down-sized by means of an interpolation method. Second, the columns of this reduced image ( $Im_1$ ) are concatenated to form a vector  $V_1$  of pixels (each pixel is characterised by a feature vector). Third, the *NO* is calculated by applying the *GS* method over the vector  $V_1$ . The *NO* obtained represents the optimal number of the image *IM*. This is because the content of  $V_1$  keeps in proportion the clusters of the original image *IM*.

2) **Method *WB*:** The *NO* is calculated using a subset  $S_{IM}$  of  $n_p$  pixels from the original image *IM*. First, the image *IM* is partitioned into  $n_B$  non-overlapping blocks of  $p_B$  pixels. Second, it is performed an over-segmentation for each block using a simplified version of *GS*. The *GS* is forced to find  $k_B$  groups and the maximum Silhouette values *MS*'s are only calculated for  $k_B$ . The  $k_B$  number is superior to the optimal number. Third, for each block, a total number of  $n_s = n_p / n_B$  pixels in proportion to the content of the clusters is selected. If a cluster has  $p_c$  pixels then it is selected from this cluster only  $ceil(p_c \times p_B / n_s)$  pixels. Fourth, the selected pixels from all the blocks form the subset  $S_{IM}$  (It contains around  $n_p$  pixels). Finally, the method *GS* is applied to the pixels of  $S_{IM}$  obtaining  $NO$  groups  $C_1 \dots C_{NO}$ . Consequently, the image *IM* has *NO* groups. The content of  $S_{IM}$  keeps in proportion the content of the original image *IM*.

### 3.2.2 Segmentation methods

The two methods *WA* and *WB* presented in the previous subsection to calculate the optimal number of groups in an image are now used to develop three new segmentation methods (*GSWA*, *GSWB*, Nyström\_ *WB*).

1) **Method GSWA:** Basically this method is performed in the following three steps:

a) Calculate the optimal number of groups  $NO$  using  $WA$ .

b) Calculate the affinity matrix  $A$  of  $IM$  using (2). In this step, we calculate a sparse version of the affinity matrix ( $A_{NN}$ ) where each pixel is only connected to the  $NN$  nearby neighbours (Shi & Malik, 1998). This step also aims to find an optimal  $NN$  ( $ONN$ ) for  $A_{NN}$ . This value is estimated by using  $NO$  (obtained in the step a) together with the Silhouette index in a similar way to the step 2 of  $GS$ .

First,  $N_v$  values for  $NN$  ( $NN_m = NN_0 + m_{NN} \times D_{NN}$ ,  $m_{NN} = 0 \dots N_v - 1$ ,  $D_{NN}$  is an integer and  $NN_0$  is the initial value) are selected. For simplicity and for each  $NN$ , the  $N$  value for the local scaling is  $N = NN$ .

Second, for each  $NN$  value,  $A_{NN}$  is calculated using only the  $NN$  nearby neighbours per pixel and (2).

Third, for each  $A_{NN}$  (and its respective matrix  $U_{NO}^{NN}$ ), the Silhouette index  $MSI_{NO}^{NN}$  for only  $NO$  groups is calculated. The Kmeans algorithm is used on the rows of  $U_{NO}^{NN}$  to find the content of the  $NO$  groups and then to calculate  $MSI_{NO}^{NN}$ .

Finally, the  $ONN$  value that corresponds to the first maximum between the  $MSI_{NO}^{NN}$ 's values is selected.

In this step, before calculating  $A_{NN}$ , the initial image  $IM$  can be down-sized. This new reduced image  $Im_2$  of size  $(S_1/F_2 \times S_2/F_2)$ , where  $F_2$  is an integer, facilitates the calculation of  $A_{NN}$  and the index  $MSI_{NO}^{NN}$ . The optimal  $NN$  value for  $Im_2$  is  $ONN_2$  (calculated as explained above). The  $ONN$  value for the original image  $IM$  will be estimated as  $ONN_2 \times F_2$ . The smaller the  $F_2$  the better the  $ONN$  value.

c) Calculate the optimal segmentation of  $IM$ . In this step, the original image  $IM$  is used together with the values  $NO$  (step a) and  $ONN$  (step b). First, the sparse matrix  $A_{ONN}$  is calculated. Then, its  $U_{NO}^{ONN}$  along with Kmeans provides the content of the  $NO$  groups. These groups form the segmentation of the image  $IM$ .

The  $SS$  is the Silhouette index used for the steps a and b.

2) **Method GSWB:** The steps of this algorithm are the following:

a) Calculate the optimal number of groups  $NO$  using  $WB$ . The respective pixels (and their feature vectors) of  $IM$  for each group from  $C_1$  to  $C_{NO}$  are selected. The Silhouette index used is  $SS$ .

b) Calculate the centroids of the feature vectors for each group of the optimal classification. The group  $C_i$  has the centroid  $c_i$ .

c) Classify each pixel of  $IM$ . A pixel of  $IM$  belongs to the group  $C_i$  if the distance between that pixel and  $c_i$  is smaller than the distance with the other centroids.

Instead of calculating the centroids, each pixel of  $IM$  can also be compared with all the pixels of the  $NO$  groups. A pixel of  $IM$  belongs to the group  $C_i$  if the distance between that pixel and other pixel (that belongs to  $C_i$ ) is smaller than the distance with the other pixels. This approach could be more suitable depending of the selected parameters for the feature vectors. For example, if the position of the

pixel is included, for feature vectors made of an unique intensity parameter per pixel, both options give the same result.

3) **Method Nyström\_WB**: This method, which makes use of *WB* together with the Nyström method (Fowlkes et al., 2004), is summarized in the following two steps:

a) Calculate the optimal number of groups *NO* using *WB*. Optionally, the respective pixels of *IM* for each group from  $C_1$  to  $C_{NO}$  (the dataset is named *DSA*) are selected. Also the pixels of *IM* that do not belong to these groups (the dataset is named *DSB*) are selected. The Silhouette index used in *WB* is *SS*.

b) Use the Nyström method (Fowlkes et al., 2004) to segment *IM*. This method classifies all the pixels of *IM* by selecting a small random subset of pixels (*RP*). From a practical point of view, it uses *RP*, the remaining pixels (*RRP*), a scaling parameter  $\sigma$  for the affinities between pixels and the number of desired groups. From the step *a*, this method can select *NO* and, optionally, the groups of pixels *DSA* and *DSB* that can be used instead of *RP* and *RRP*.

The Nyström method provides a matrix of orthogonal eigenvectors since the affinity matrix (2) is definite positive (see details in (Fowlkes et al., 2004)). This orthogonalization gives stability to the classification outcome.

We distinguish the Nyström method without orthogonalization (Nyström\_WB\_WO) and the Nyström method with orthogonalization (Nyström\_WB\_O) as in (Chen, Song, Bai, Lin, & Chang, 2011). The Nyström\_WB\_WO outcome is variable depending on the *RP* chosen. Using different *RP*'s, the best result is selected by means of the highest *SS* index. In this case, this index applies between the original image and each obtained segmentation.

#### 4. Testing the algorithm

In this section, first the *GS* algorithm is tested using synthetic data. Then, the methods *GSWA*, *GSWB* and Nyström\_WB are tested on real data by means of an image segmentation problem.

##### 4.1 Synthetic data points

Fig. 1 shows the groups found using 6 datasets (Zelnik-manor & Perona, 2004) and the *GS* algorithm. The *NO* found for each dataset matches the number of meaningful groups.

The *HC* algorithm has been used for the unsupervised classifications of *GS*. Table 1 shows for each dataset (identified by its number in Fig. 1) the arrays  $VMSI^5$ ,  $VNO^5$ ,  $GVMSI$  and  $GVNO$  (for  $N_0=5$  and  $D=5$ ). The optimal  $MS^5$  for  $VMSI^5$ ,  $GMS$  for  $GVMSI$  and their respective  $NO^5$  and *NO* are highlighted in bold. The optimal *GMS* in  $GVMSI$  allows to find *ON* and the matrix *OU*.

In each dataset, the optimal  $NO$  selected in  $GVNO$  is also the most often repeated number in that vector. It is unique for all the datasets except for the fourth dataset. In this dataset, as it can be seen in Fig. 1, the  $NO^5=6$  and the optimization process allows to find  $NO=5$ .

The Silhouette indices  $AS$  and  $SS$  have found the same number of meaningful groups. But as stated in the subsection 2.4  $SS$  needs less computational resources.

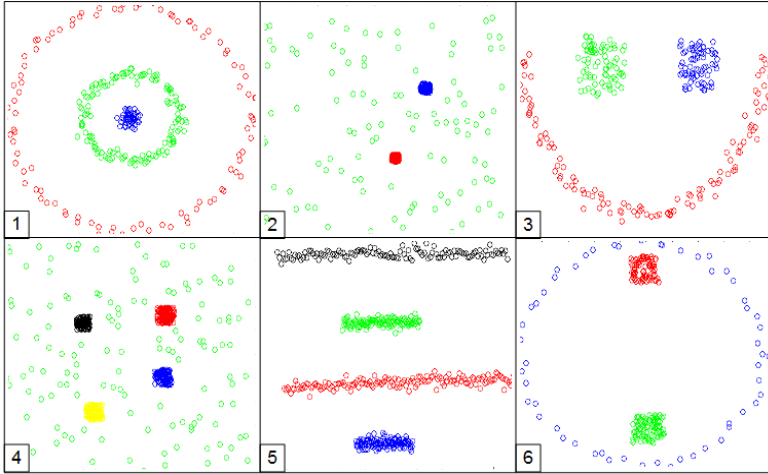


Fig. 1: Groups found for 6 datasets using the  $GS$  algorithm.

**Table 1**  
Arrays *GVMSI* and *GVNO* using *AS* and *SS* for 6 datasets to select their *NO*, *ON* and *OU*

Dataset 1 and AS					Dataset 1 and SS				
<i>VMSF</i>	$MSI_2^5=0,84$	<b><math>MSI_3^5=0,89</math></b>	$MSI_4^5=0,84$		<i>VMSF</i>	$MSI_2^5=0,88$	<b><math>MSI_3^5=0,92</math></b>	$MSI_4^5=0,88$	
<i>VNO</i> <sup>5</sup>	3	3	4		<i>VNO</i> <sup>5</sup>	4	3	4	
<i>GVMSI</i>	$MSI_3^5=0,89$	$MSI_3^{15}=0,93$	<b><math>MSI_3^{20}=0,944</math></b>	$MSI_3^{25}=0,943$	<i>GVMSI</i>	$MSI_3^5=0,92$	$MSI_3^{15}=0,95$	<b><math>MSI_3^{20}=0,958</math></b>	$MSI_3^{25}=0,957$
<i>GVNO</i>	3	3	3	3	<i>GVNO</i>	3	3	3	3
Dataset 2 and AS					Dataset 2 and SS				
<i>VMSF</i>	$MSI_2^5=0,95$	<b><math>MSI_3^5=0,915</math></b>	$MSI_4^5=0,888$		<i>VMSF</i>	$MSI_2^5=0,97$	<b><math>MSI_3^5=0,94</math></b>	$MSI_4^5=0,92$	
<i>VNO</i> <sup>5</sup>	3	3	4		<i>VNO</i> <sup>5</sup>	3	3	4	
<i>GVMSI</i>	$MSI_3^5=0,915$	$MSI_3^{10}=0,947$	<b><math>MSI_3^{15}=0,95</math></b>	$MSI_3^{20}=0,94$	<i>GVMSI</i>	$MSI_3^5=0,94$	$MSI_3^{10}=0,96$	<b><math>MSI_3^{15}=0,965</math></b>	$MSI_3^{20}=0,963$
<i>GVNO</i>	3	3	3	3	<i>GVNO</i>	3	3	3	3
Dataset 3 and AS					Dataset 3 and SS				
<i>VMSF</i>	$MSI_2^5=0,79$	<b><math>MSI_3^5=0,85</math></b>	$MSI_4^5=0,73$		<i>VMSF</i>	$MSI_2^5=0,85$	<b><math>MSI_3^5=0,89</math></b>	$MSI_4^5=0,79$	
<i>VNO</i> <sup>5</sup>	3	3	4		<i>VNO</i> <sup>5</sup>	3	3	4	
<i>GVMSI</i>	$MSI_3^5=0,85$	$MSI_3^{15}=0,91$	<b><math>MSI_3^{20}=0,941</math></b>	$MSI_3^{25}=0,94$	<i>GVMSI</i>	$MSI_3^5=0,89$	$MSI_3^{20}=0,954$	<b><math>MSI_3^{25}=0,956</math></b>	$MSI_3^{30}=0,92$
<i>GVNO</i>	3	3	3	3	<i>GVNO</i>	3	3	3	3
Dataset 4 and AS					Dataset 4 and SS				
<i>VMSF</i>	$MSI_3^5=0,85$	$MSI_4^5=0,87$	$MSI_5^5=0,84$	<b><math>MSI_6^5=0,82</math></b>	<i>VMSF</i>	$MSI_3^5=0,89$	$MSI_4^5=0,91$	$MSI_5^5=0,88$	<b><math>MSI_6^5=0,87</math></b>
<i>VNO</i> <sup>5</sup>	5	5	6	6	<i>VNO</i> <sup>5</sup>	5	5	6	6
<i>GVMSI</i>	$MSI_6^5=0,82$	<b><math>MSI_6^{10}=0,90</math></b>	$MSI_6^{15}=0,898$	$MSI_6^{20}=0,895$	<i>GVMSI</i>	$MSI_6^5=0,87$	<b><math>MSI_6^{10}=0,9275</math></b>	$MSI_6^{15}=0,927$	
<i>GVNO</i>	6	5	5	5	<i>GVNO</i>	6	5	5	
Dataset 5 and AS					Dataset 5 and SS				
<i>VMSF</i>	$MSI_2^5=0,86$	$MSI_3^5=0,87$	<b><math>MSI_4^5=0,91</math></b>	$MSI_5^5=0,82$	<i>VMSF</i>	$MSI_2^5=0,89$	$MSI_3^5=0,90$	<b><math>MSI_4^5=0,93</math></b>	$MSI_5^5=0,86$
<i>VNO</i> <sup>5</sup>	2	4	4	6	<i>VNO</i> <sup>5</sup>	2	4	4	6
<i>GVMSI</i>	$MSI_4^5=0,91$	$MSI_4^{30}=0,94$	<b><math>MSI_4^{60}=0,964</math></b>	$MSI_4^{65}=0,963$	<i>GVMSI</i>	$MSI_4^5=0,93$	$MSI_4^{30}=0,95$	<b><math>MSI_4^{60}=0,973</math></b>	$MSI_4^{65}=0,972$
<i>GVNO</i>	4	4	4	4	<i>GVNO</i>	4	4	4	4
Dataset 6 and AS					Dataset 6 and SS				
<i>VMSF</i>	$MSI_2^5=0,86$	<b><math>MSI_3^5=0,87</math></b>	$MSI_4^5=0,83$		<i>VMSF</i>	$MSI_2^5=0,89$	<b><math>MSI_3^5=0,90</math></b>	$MSI_4^5=0,87$	
<i>VNO</i> <sup>5</sup>	3	3	4		<i>VNO</i> <sup>5</sup>	3	3	4	
<i>GVMSI</i>	$MSI_3^5=0,87$	<b><math>MSI_3^{10}=0,89</math></b>	$MSI_3^{15}=0,87$		<i>GVMSI</i>	$MSI_3^5=0,90$	<b><math>MSI_3^{10}=0,917</math></b>	$MSI_3^{15}=0,90$	
<i>GVNO</i>	3	3	3		<i>GVNO</i>	3	3	3	

#### 4.2 Real data: image segmentation

In this subsection, the goal is to segment some grey images grouping pixels with similar intensity. Each pixel is characterised by a feature vector of one parameter (its intensity).

The distance between pixels for the affinity matrix (2) is defined by the difference of their intensities (6):

$$d(x, y) = I(x) - I(y) \quad (6)$$

**Method GSWA:** Fig. 2 shows a grey image  $IM_1$  of size 160x160 (Boy, 2016) and its segmentations for different values of  $NN$  obtained using the  $GSWA$  algorithm.

For the step  $a$ , the image  $IM_1$  is down-sized to a size 40x40. The  $NO$  obtained for  $IM_1$  is  $NO=4$ .

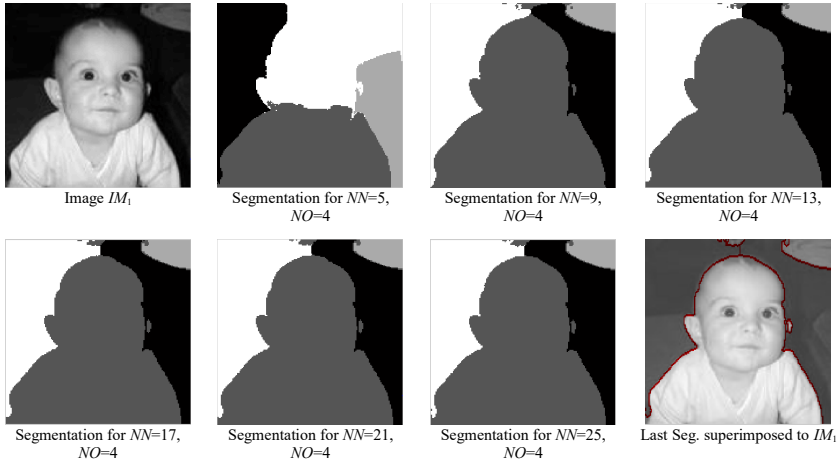


Fig. 2 Gray image  $IM_1$  used to test  $GSWA$  and its segmentations for different values of  $NN$ . The last image shows the segmentation for  $NN=25$  superimposed to  $IM_1$ .

For the step  $b$  of  $GSWA$ , it has been selected a value of  $F_2=2$ . The  $ONN$  found is  $ONN=26$  with  $ONN_2 = 13$  (see Table 2).

**Table 2**

Selection of  $ONN_2$  for  $Im_2$  with  $F_2=2$  and  $NO=4$

$MSI_4^{NN}$	$MSI_4^5$	$MSI_4^7$	$MSI_4^9$	$MSI_4^{11}$	<b><math>MSI_4^{13}</math></b>	$MSI_4^{17}$	$MSI_4^{21}$	$MSI_4^{25}$
Value	0.6626	0.7515	0.7673	0.7708	<b>0.7711</b>	0.7711	0.7698	0.7689

For the step  $c$  of  $GSWA$ , the sparse matrix  $A_{25}$  of the original image  $IM_1$  has been calculated using the values  $NO=4$  and  $ONN=25$  (an odd number close to 26) obtained in steps  $a$  and  $b$  of  $GSWA$ . The value of  $ONN=25$  instead of 26 facilitates the calculation of the sparse matrix. The matrix of eigenvectors  $U_4^{25}$

and Kmeans provide the content of the 4 groups. The  $ON$  used for the local scaling is  $ON=25$ .

Table 3 shows the  $MSI$  values for different values of  $NN$  using the image  $IM_1$ . In Fig. 2, it is not possible distinguish significant differences between the segmentation for  $ONN=25$  and the one for  $NN=21$  (the values of Table 3  $MSI_4^{25}$  and  $MSI_4^{21}$  are also very similar). In this case, the choice of  $ONN$  does not need to be accurate. A value of  $ONN=21$  (instead of 25) reduces the number of calculations without changing significantly the quality of the  $IM_1$  segmentation.

**Table 3**

$MSI$  values for different values of  $NN$  using the image  $IM$  and  $NO=4$

$MSI_4^{NN}$	$MSI_4^5$	$MSI_4^9$	$MSI_4^{13}$	$MSI_4^{17}$	$MSI_4^{21}$	$MSI_4^{25}$
Value	0.5910	0.7569	0.8142	0.8292	0.8329	<b>0.8335</b>

In this example it has been possible to find an optimal value for  $NN$  but, in general, this optimal value can be high and the step  $c$  of  $GSWA$  has a very high computational cost. When this happens, the solution is to select the highest possible value for  $NN$ . It is expected to find an image segmentation with an acceptable quality.

For simplicity, it is selected  $N=NN$ . No differences were found for different values of  $N<NN$ . However, changes can occur for different values of  $NN$ . If the size of an image either increases or decreases, the optimal  $NN$  value for this image changes in a proportional way.

In addition to the need of large computational resources for high values of  $NN$ , the method  $GSWA$  shows other important drawback: the emergence of different segments in uniform regions (see the separation over the head in Fig. 2).

The next methods overcome these problems by means of a scalable approach based on the use of  $WB$ .

**Method  $GSWB$ :** Fig. 3 shows the segmentations of  $IM_1$  for different values of  $n_p$  obtained using the  $GSWB$  algorithm.

For the step  $a$ , it has been selected for  $WB$ : the image  $IM_1$  (with size  $160 \times 160$ ) of Fig.2,  $n_p=[700, 1300, 2000]$  pixels,  $n_B=64$  blocks and  $k_B=8$ . For the local scaling  $N=5$  and  $D=5$ . Fig. 3 shows the over-segmentation of the 64 blocks. The  $WB$  applied to the subset  $S_{IM}$  has found  $NO=3$  for all the  $n_p$  values. The outcome of the segmentation is very similar in the three cases due to the stability of the centroids. The algorithm is fairly quick although the calculation of  $NO$  in  $WB$  is the slower part.

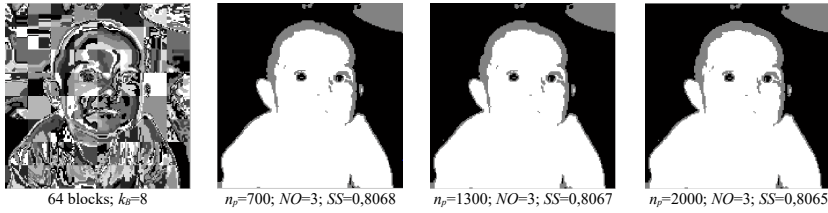


Fig. 3 Over-segmentation of the blocks of  $IM_1$  and its segmentations using  $GSWB$  for  $n_p=[700, 1300, 2000]$  pixels.

**Method Nyström\_WB:** Fig. 4 shows a set of  $IM_1$  segmentations using the  $Nyström\_WB\_WO$  for different random subset of pixels and  $NO=3$ . The segmentations found with the best quality are represented. Some segmentations are similar but the entire set presents a significant variability. The outcome of  $Nyström\_WB\_WO$  is not unique.

The index  $SS$  can be used to select the best segmentation. This is very similar to the segmentation obtained using the  $GSWB$  method. As Fig. 4 shows, the more  $RP$  does not mean better quality.



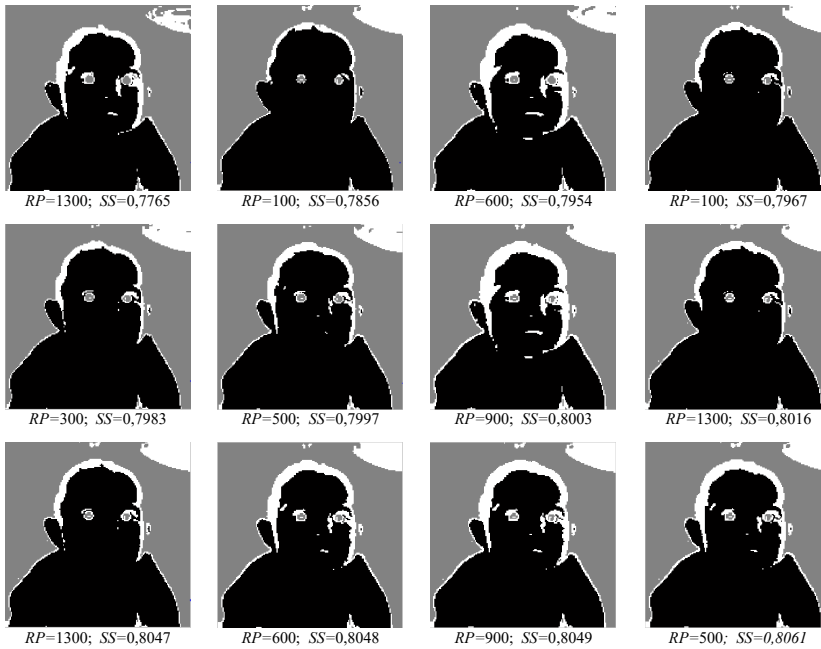


Fig. 4 Segmentations of  $IM_1$  using Nyström\_WB\_WO for different  $RP$  and  $NO=3$ . Their quality varies from  $SS=0,776$  to  $SS=0,806$ .

Fig. 5 shows the outcome of the Nyström\_WB\_WO using the subsets of pixels  $DSA$  and  $DSB$  obtained in  $WB$  for  $n_p=[700, 1300, 2000]$  and  $NO=3$ . The variability of the segmentation for different subsets of pixels is significant. This option does not produce better results than using random subsets of pixels and consequently, does not improve the Nyström\_WB\_WO.



Fig. 5 Segmentations of  $IM_1$  using Nyström\_WB\_WO and the subsets of pixels  $DSA$  and  $DSB$  obtained in  $WB$  for  $n_p=[700, 1300, 2000]$ .

Fig. 6 shows the  $IM_1$  segmentation using the Nyström\_WB\_O for  $RP=300$ . The four trials have found the same segmentation. For  $RP=300$  the Nyström\_WB\_O outcome is steady. This segmentation is similar to the best of Nyström\_WB\_WO and the one of GSWB. The orthogonalization process in Nyström\_WB\_O needs large computational resources for mid/high  $RP$  values. Consequently, it is not appropriate to use the sets of pixels  $DSA$  and  $DSB$ .

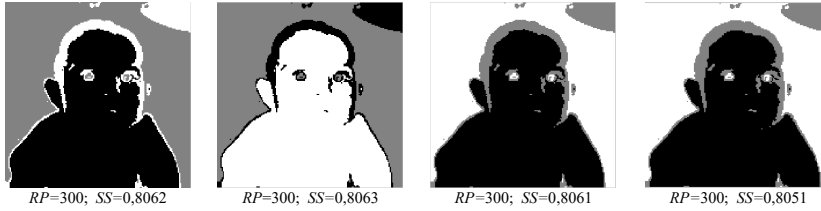


Fig. 6 Segmentations of  $IM_1$  using Nyström\_WB\_O for four trials.

Fig. 7 and 8 show the segmentations of four images ( $IM_2$ ,  $IM_3$ ,  $IM_4$ ,  $IM_5$ ) from (Berkeley, 2016) using GSWB and Nyström\_WB. Both approaches find similar results.

Image	GSWB	Nyström_WB_WO	Nyström_WB_O
 $IM_2$	 $n_p=700; NO=4; SS=0.7994$	 $RP=700; NO=4; SS=0,80$	 $RP=300; NO=4; SS=0,7947$
 $IM_3$	 $n_p=700; NO=3; SS=0,9015$	 $RP=700; NO=3; SS=0,9017$	 $RP=300; NO=3; SS=0,9012$

Fig. 7 Segmentations of two images from (Berkeley, 2016) using GSWB and Nyström\_WB

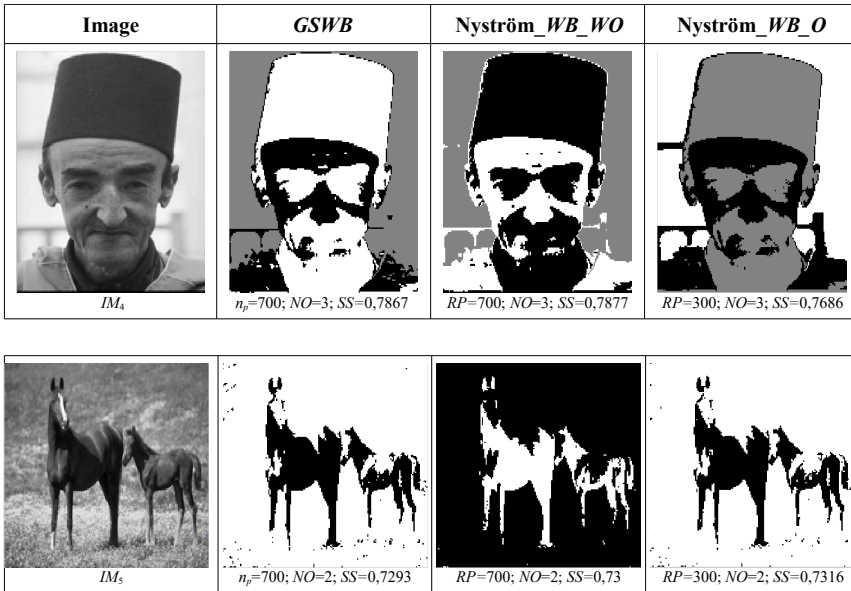


Fig. 8 Segmentations of two images from (Berkeley, 2016) using *GSWB* and Nyström\_ *WB*

## 5. Discussion

As it has been shown in Section 4, the *GS* algorithm has correctly calculated *NO* when using synthetic data since meaningful groups have been found.

The *GS* algorithm works directly with the new representations of the data points obtained by using the *SC* algorithm. This allows extending the application of *VID* used in other clustering algorithms. The use of the Silhouette index *SS* in *GS* is motivated by the fact of its robustness, easy implementation and its low computational complexity. Moreover local scaling and *SS* are mutually beneficial. Local scaling improves the separation between clusters and consequently it is good for the Silhouette index. At the same time Silhouette allows finding an optimal neighbour of the local scaling through which the *GS* algorithm stops. The first local maximum of the *MS*'s allows stopping the search of *NO*. This criterion has been tested in all our experiments which have reported good results.

Concerning the image segmentation, the methods *GSWA*, *GSWB* and Nyström\_ *WB* have been analysed. The *GSWA* uses the *WA* method. In *WA*, the *GS* algorithm is applied to a full reduced image. The main drawbacks identified for *GSWA* include the large computational needs and the over-segmentation of uniform regions.

The *GSWB* and *Nyström\_WB* methods do not have these problems. They use the *WB* method based on a scalable approach whereby it is possible to segment any image without huge computational needs.

The *Nyström\_WB* and *GSWB* are very different but they find similar results. Their small differences are not significant. This similarity is a validation of *GSWB*.

Once *WB* has calculated *NO*, *GSWB* is very quick to segment an image. The *Nyström\_WB* method is slower.

The over-segmentation of the blocks in *WB* imposes a minimum number of pixels for *GSWB*. The *GSWB* outcome is unique and of good quality due to the optimization process in *WB*. This result does not vary significantly when the number of pixels  $n_p$  changes.

The *Nyström\_WB\_WO* outcome is variable depending on the group of *RP* pixels initially chosen. It is necessary to use the *SS* index between the original image and the segmentations (obtained using various trials of *RP*) to select the best result.

The *Nyström\_WB\_O* uses a unique number *RP* of pixels whereby the segmentation is steady for any trial. Due to the orthogonalization process, the use of *Nyström\_WB\_O* is more restrictive than *Nyström\_WB\_WO*.

For an image, the highest *SS* index of *Nyström\_WB\_WO* can be compared with the *SS* index of the *GSWB* method and the one of *Nyström\_WB\_O*. They are similar so the *SS* index also allows validating quantitatively the *GSWB* method (see the images of Fig. 7 and 8).

One might think in a method to calculate *NO* by combining *Nyström* and *SS*. In some experiments the results were not correct. The key point is that *SS* provides good results in combination with the local scaling and the *Nyström* method only uses an unique scaling parameter.

The quality of the segmentation depends on the quality of the original image and also on the pixels distance chosen. The distance (6) is quite simple and consequently, is expected to find, for example, over-segmentations such as the dome of the church (see the first image of Fig. 7). On the other hand, in the last image of Fig. 8, the horses are segmented but due to (6) some parts of their body are fused with the environment.

The segmentation methods have been tested using *NO* but this does not prevent to use these methods to work with other number of groups that are non-optimal. For example, in the *GSWB* method, it is possible to cluster the subset  $S_{IM}$  into the desired number of groups instead of *NO*.

The *GSWB* method has been tested by means of some image segmentations. The images contain a large set of pixels and their segmentation can easily be observed in a compact way. Image segmentation by means of *GSWB* provides a good example of how *GS* can be applied for any large dataset.

Compared to (Zelnik-manor & Perona, 2004) (*ZP*) and (Xiang & Gong, 2008) (*XG*) our work presents significant differences. As it was mentioned in section 1,

*GS* works directly with the new representation of the data points unlike *ZP* and *XG* that initially work with eigenvectors.

We can also classify these methods in a different way. The *GS* and *ZP* obtain the optimal number of clusters preserving the full information of the original data while *XG* works selecting the most significant information according to a specific criterion.

The *GS* algorithm and *ZP* use the local scaling concept and both find the same result for synthetic data as the 6 datasets of Fig. 1. However, *ZP* is sensitive to noise and for real data this method cannot work correctly. Due to the properties of the Silhouette index, the *GS* algorithm is more efficient against noise. The *ZP* algorithm also needs to define a cost function. This fact is not necessary in *GS*. In addition, *GS* searches for an optimal local scaling.

Fig. 9 shows some examples of image segmentation using the *ZP* algorithm with (6). Unlike the synthetic data, the *ZP* method is less effective with images for both the optimal number the clusters and their contents. For example, the image segmentations of  $IM_2$  and  $IM_3$  are rather poor. In  $IM_1$  and  $IM_4$ , some over-segmentations appear.

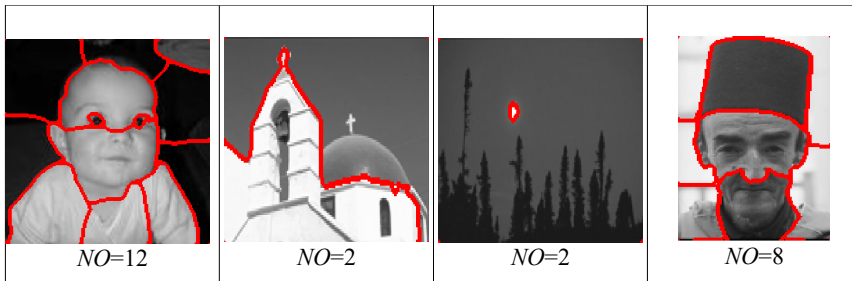


Fig. 9 Segmentation of the images  $IM_1$ ,  $IM_2$ ,  $IM_3$  and  $IM_4$  using the *ZP* method (Zelnik-Manor Resources, 2016).

Unlike *GS*, the *XG* algorithm does not take advantage of the local scaling concept. It uses an eigenvector selection to improve spectral clustering results. It measures the relevance of an eigenvector according to how well it can separate a dataset into different clusters. *XG* measures how informative/relevant each eigenvector is and eliminates the less relevant eigenvectors. The set of relevant eigenvectors makes clustering results less susceptible to noise in the data. However, these eigenvectors cannot always represent the data structure. For example, as established in (Xiang & Gong, 2008), the algorithm *XG* is able to deal with unevenly distributed data sets as long as the size difference between clusters is not too extreme. This does not happen with *ZP* and *GS* because they make use of the local scaling concept and can cluster data containing multiple scales (see for example the datasets 2 and 4 of Fig.1).

The *XG* algorithm estimates the optimal number of clusters from the relevant eigenvectors with the help of the Bayesian Information Criterion (Schwarz, 1978).

The main characteristics of the *GS* algorithm are summarized as follows:

- The optimal number of clusters is directly calculated from the new representations of the data points in *SC*.
- It combines the concept of local scaling with the Silhouette index.
- By means of the Silhouette index, *GS* is robust against noise.
- The Silhouette index also allows finding an optimal local scaling.
- The local scaling concept improves the behaviour of the Silhouette index and is especially useful when the data contains multiple scales.
- The Simplified Silhouette Index reduces significantly the computational needs.
- The algorithm can be used for large datasets as for example image segmentation.

The results obtained in this paper for synthetic data and images allow showing the behaviour of the *GS* algorithm. The synthetic data are used to check that *GS* is a useful method to calculate the optimal number of clusters. The images are used to show how to use *GS* for large amount of data.

The clustering results can be evaluated in a visual way for both synthetic data and images. The index *SS* allows also to check quantitatively the image segmentations. This index together with the visual inspection provides a complete evaluation of the image segmentations in a compact way. Consequently, it is not necessary to use other evaluation metrics as *ARI* (Hubert & Arabie, 1985) or *NMI* (Strehl & Ghosh, 2002) for a quantitative evaluation of the results.

## 6. Conclusion

A simple and effective new method *GS* to find the optimal number of clusters *NO* in a dataset using the spectral clustering algorithm has been presented. It combines the Silhouette validity index together with the concept of local scaling. The *GS* algorithm has been tested using synthetic data.

Image segmentation has been used to show how to apply *GS* for a large dataset. Three new segmentation methods (*GSWA*, *GSWB*, *Nyström\_WB*) have been described. In general, the *GSWA* algorithm is less effective than *GSWB* and *Nyström\_WB*. Once *NO* has been calculated by means of *WB*, *GSWB* and *Nyström\_WB* find similar image segmentations but *GSWB* is much quicker. The Silhouette index *SS* is also used to find the best segmentation for the *Nyström\_WB\_WO*. Furthermore, it allows validating quantitatively the *GSWB* algorithm.

## Acknowledgements

This work was supported in part by the Spanish Ministry of Economy and Competitiveness under Project DPI2011-27818-C02-02, DPI2012-31303, DPI2014-55932-C2-2-R, ENE2015-64914-C3-1-R and ENE2015-64914-C3-2-R and FEDER funds.

## References

- Boy Image. (2016). <http://timotheecour.com/software/ncut/ncut.html>. Accessed 24/02/2016
- Berkeley Dataset. (2016). <https://www.eecs.berkeley.edu/Research/Projects/CS/vision/bsds/BSDS300/html/dataset/images.html>. Accessed 24/02/2016
- Calinski, T., & Harabasz, J. (1974). A Dendrite Method for Cluster Analysis. *Communications in Statistics - Simulation and Computation*, 3(1), 1–27. <http://doi.org/10.1080/03610917408548446>
- Chen, W. Y., Song, Y., Bai, H., Lin, C. J., & Chang, E. Y. (2011). Parallel spectral clustering in distributed systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(3), 568–586. <http://doi.org/10.1109/TPAMI.2010.88>
- Davies, D. L., & Bouldin, D. W. (1979). A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1(2), 224–227. <http://doi.org/10.1109/TPAMI.1979.4766909>
- Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1), 1–38. <http://doi.org/10.1111.133.4884>
- Dunn, J. C. (1974). Well-Separated Clusters and Optimal Fuzzy Partitions. *Journal of Cybernetics*, 4(1), 95–104. <http://doi.org/10.1080/01969727408546059>
- Everitt, B., Landau, S., Leese, M., & Stahl, D. (2011). *Cluster analysis. Wiley series in probability and statistics*.
- Fowlkes, C., Belongie, S., Chung, F., & Malik, J. (2004). Spectral Grouping Using the Nyström Method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2), 214–225. <http://doi.org/10.1109/TPAMI.2004.1262185>
- Halkidi, M., & Vazirgiannis, M. (2001). Clustering Validity Assessment : Finding the optimal partitioning of a data set. In *In Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on* (pp. 187–194). <http://doi.org/10.1109/ICDM.2001.989517>

- Hruschka, E. R. & Covões, T. F. (2005). Feature Selection for Cluster Analysis: an Approach Based on the Simplified Silhouette Criterion. *CIMCA/LAWTIC, IEEE Computer Society*, 32–38.
- Hubert, L. & Arabie P. (1985). Comparing partitions. *J CLASSIF*, 2(1):193–218.
- Jain, A. K., Murty, M. N., & Flynn, P. J. (1999). Data clustering: A review. *ACM Computing Surveys*, 31(3), 264–323.
- Luxburg, U. Von. (2007). A Tutorial on Spectral Clustering A Tutorial on Spectral Clustering. *Technical Report*, 17(March), 395–416. <http://doi.org/10.1007/s11222-007-9033-z>
- MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, 233(233), 281–297.
- Ng, A. Y., Jordan, M. I., & Weiss, Y. (2001). On Spectral Clustering: Analysis and an algorithm. *Advances in Neural Information Processing Systems*, 849–856. <http://doi.org/10.1.1.19.8100>
- Rokach, L., & Maimon, O. (2005). Clustering methods. In *Data mining and knowledge discovery handbook* (pp. 321–352). [http://doi.org/10.1007/0-387-25465-X\\_15](http://doi.org/10.1007/0-387-25465-X_15)
- Rousseeuw, P. J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20, 53–65. [http://doi.org/10.1016/0377-0427\(87\)90125-7](http://doi.org/10.1016/0377-0427(87)90125-7)
- Shi, J. S. J., & Malik, J. (1998). Motion segmentation and tracking using normalized cuts. *Sixth International Conference on Computer Vision IEEE Cat No98CH36271*. <http://doi.org/10.1109/ICCV.1998.710861>
- Schwarz G. (1978). Estimating the dimension of a model, *Annals Statistics*, 6(2): 461–464.
- Strehl A. & Ghosh J. (2002). Clustering Ensembles: a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research* 3, 583-617.
- Tung, F., Wong, A., & Clausi, D. A. (2010). Enabling scalable spectral clustering for image segmentation. *Pattern Recognition*, 43(12), 4069–4076. <http://doi.org/10.1016/j.patcog.2010.06.015>
- Xiang, T., & Gong, S. (2008). Spectral clustering with eigenvector selection. *Pattern Recognition*, 41(3), 1012–1029. <http://doi.org/10.1016/j.patcog.2007.07.023>



### *Capítulo 6: ART.3*

Xu, R., & Wunsch, D. (2005). Survey of clustering algorithms. *IEEE Transactions on Neural Networks*, 16, 645-678. <http://doi.org/10.1109/TNN.2005.845141>

Zelnik-Manor, L., Perona, P., (2004). Self-tuning spectral clustering. *Advances in Neural Information Processing Systems* 17, 2, 1601–1608. <http://doi.org/10.1.1.84.7940>

Zelnik-Manor Resources. (2016). <http://lihi.eew.technion.ac.il/resources/>. Accessed 4/07/2016.



## CAPÍTULO 7


### ART.4: AN UNSUPERVISED METHOD TO DETERMINE THE OPTIMAL NUMBER OF INDEPENDENT COMPONENTS


Resumen:

- *Revista:* Expert Systems with Applications
- *Current Impact Factor:* 2.981; *5-Year Impact Factor:* 2.879
- *Selección de la primera página del artículo publicado:*

Expert Systems With Applications 75 (2017) 56–62

Contents lists available at [ScienceDirect](#)

 **Expert Systems With Applications**  
journal homepage: [www.elsevier.com/locate/eswa](http://www.elsevier.com/locate/eswa)

 CrossMark

#### An unsupervised method to determine the optimal number of independent components

Angel Mur<sup>a</sup>, Raquel Dormido, Natividad Duro, Daniel Mercader

*Department of Computer Sciences and Automatic Control, UNED, Juan del Rosal 16 - 28040 Madrid, Spain*

---

**ARTICLE INFO**

*Article history:*  
Received 20 October 2016  
Revised 17 January 2017  
Accepted 18 January 2017  
Available online 20 January 2017

*Keywords:*  
Distance correlation  
ICA  
JADE algorithm  
Linear correlation  
Optimal number of independent components

**ABSTRACT**

In this paper, we present a new method to determine the optimal number of independent components after applying an Independent Component Analysis (ICA) to a set of mixed signals. The proposed method, called Linear Correlations between Components (LCC), uses the JADE algorithm to calculate the independent components. The LCC method allows to automatically select the optimal number of independent components in an unsupervised way without any previous knowledge. It has been tested using synthetic mixed signals where the number of pure (or independent) signals is known. This method is very simple, fast and easy to implement.

© 2017 Elsevier Ltd. All rights reserved.

## An unsupervised method to determine the optimal number of independent components

Angel Mur<sup>a</sup>, Raquel Dormido<sup>a</sup>, Natividad Duro<sup>a</sup>, Daniel Mercader<sup>a</sup>

<sup>a</sup> Department of Computer Sciences and Automatic Control, UNED, Juan del Rosal 16 - 28040 Madrid, Spain, a.r.m.g@outlook.fr, raquel@dia.uned.es, nduro@dia.uned.es, mercaderd@yahoo.es

### ABSTRACT

In this paper, we present a new method to determine the optimal number of independent components after applying an Independent Component Analysis (*ICA*) to a set of mixed signals. The proposed method, called Linear Correlations between Components (*LCC*), uses the *JADE* algorithm to calculate the independent components. The *LCC* method allows to automatically select the optimal number of independent components in an unsupervised way without any previous knowledge. It has been tested using synthetic mixed signals where the number of pure (or independent) signals is known. This method is very simple, fast and easy to implement.

**Keywords:** Distance Correlation, *ICA*, *JADE* algorithm, Linear Correlation, Optimal Number of Independent Components

### 1. Introduction

Given a set of mixed signals that have been created by combining a set of pure signals in unknown proportions, the Independent Components Analysis (*ICA*) (Bouveresse, & Rutledge, 2016; A Hyvärinen & Oja, 2000) is a blind-source separation method that enables the extraction of the pure signals, as well as their proportions, from the set of mixed signals. *ICA* has been applied in many domains in which signals have to be analysed (Hao et al., 2009; He, Clifford, & Tarassenko, 2006; Krishnaveni, Jayaraman, Manoj Kumar, Shivakumar, & Ramadoss, 2005; Wang, Ding, & Hou, 2008). In particular, it is used to separate significant components from signals and remove artefacts.

*ICA* is based on the construction of the Independent Components (*IC*). The procedure for determining the optimal number of *IC*'s,  $k$ , is an important issue when developing an *ICA* model. The value of  $k$  corresponds to the number of pure signals where each of them explains an independent phenomenon. In general, when the number of desired *IC*'s,  $NIC$ , is smaller than the optimal one, some significant components are mixed together in the small number of extracted *IC*'s. On the other hand, if a  $NIC$  bigger than  $k$  is required, some of the significant components are decomposed into subcomponents (it means that not all the components are independent). In both cases, the components obtained neither represent nor explain correctly the independent phenomena. Therefore, it is

necessary to use a validation method to determine  $k$ . This is the main goal of this paper.

Some methods make use of some theoretical knowledge of the pure signals to determine  $k$ , such as the Amari index (Rutledge & Jouan-Rimbaud Bouveresse, 2013). However, in practice this information is not available and consequently any method to determine  $k$  should work without a priori knowledge. This fact involves the need to find an unsupervised method to determine the optimal number of independent components when analyzing a set of mixed signals.

Ref. (Bouveresse, & Rutledge, 2016) shows a review of the most interesting algorithms developed to find  $k$ : the Durbin-Watson criterion (Rutledge & Barros, 2002), the *ICA\_by\_Blocks* (Jouan-Rimbaud Bouveresse, Moya-González, Ammari, & Rutledge, 2012), the *Random\_ICA\_by\_Blocks*, the *RV\_ICA\_by\_Blocks* and the *ICA\_corr\_Y*. Unlike *ICA\_corr\_Y*, the first four methods do not require any specific prior knowledge. The Durbin-Watson criterion can only be used in structured signals although the other methods can be applied to any type of data. The methods that use blocks of data need to pay particular attention in selecting representative and comparable data blocks. Furthermore, these “blocks methods” cannot find  $k$  when the number of mixed signals is less than  $2 \times k$ .

In addition to these methods, there is another approach based on the Principal Components Analysis (*PCA*) (Semmlow, 2004; Jolliffe, 2002) applied to the mixed signals. This algorithm considers the optimal number of *IC*'s equal to the number of significant Principal Components. This method is simple but it is based on hypotheses that are not always valid. Moreover, the optimal number of *IC*'s is chosen from a scree plot. This plot is a descending curve representing the eigenvalues vs the Principal Component index. The optimal number of *IC*'s is the value at which the eigenvalues start to level off.

In the present work, *JADE* is the selected *ICA* algorithm (Cardoso & Souloumiac, 1993). *JADE* consists of an eigenmatrix decomposition of a higher-order cumulant tensor. The cumulants give a measure of the non-Gaussianity of the components. For each *NIC*, *JADE* provides a stable result. This algorithm starts by restricting the operation of *JADE* to the *NIC* first principal components obtained from a *PCA* (Cardoso Resources, 2016).

In this paper, a simple and effective method to calculate  $k$  named Linear Correlations between Components (*LCC*) is shown. This algorithm uses *JADE* and it takes advantages of both the concept of independence and the fact that a decomposed *IC* appears when *NIC* is superior to  $k$ . The *LCC* method allows to automatically select  $k$  in an unsupervised way.

Unlike *LCC*, every method described above needs at least a specific condition to find  $k$  that reduces its capacity of being generic. In general, these methods require a graphical representation to select  $k$ .

On the other hand, the main characteristics of *LCC* are the following: it is an unsupervised method, it determines  $k$  automatically, it does not need the support

of a graphical representation, there is no restriction with respect to the type of pure signals, the mixed signals do not need to be ordered in blocks and it can calculate  $k$  when the number of mixed signals is inferior to  $2 \times k$ . However, as any method, it needs a number of mixed signals superior to  $k$ .

The only limitation (similar to others methods) comes from the computational resources that *JADE* needs. Depending on the available memory and *CPU* time, *JADE* cannot create a very large number of components (Cardoso Resources, 2016).

In practice, the  $k$  found cannot be validated by using external information. One way to evaluate the result is to compare the results of different algorithms. The *LCC* is a different method and can contribute to the evaluation process. However, *LCC* is robust enough and its results can be trusted.

In Section 2, *ICA* and some basic concepts used in *LCC* are reviewed. In Section 3, the proposed method *LCC* is described. In Section 4, the *LCC* and other methods are tested using synthetic data. Finally, in Sections 5 and 6, the discussion and conclusions of the paper are respectively presented.

## 2. Background

This section explains the *ICA* algorithm and reviews the concepts of independence and correlation between two random variables.

### 2.1 *ICA*

Given a set of  $n$  mixed signals formed by combining  $k$  independent signals with  $p$  samples each, its *ICA* model is defined as  $M=A \times S$ , where  $M$  is a  $n \times p$  data matrix,  $A$  is a  $n \times k$  mixing matrix and  $S$  is a  $k \times p$  matrix of independent signals.

The objective of *ICA* is to calculate  $A$  and  $S$  knowing only  $M$ . *ICA* does not need any knowledge concerning the nature of the source signals or their proportions. To estimate  $A$ , *ICA* requires the pure signals in  $S$  to be truly independent and non-Gaussian. Both conditions are usually met when the sources are real signals.

The independence in *ICA* can be reached by maximizing the non-Gaussianity of the components or by minimizing the mutual information (Wang et al., 2008). Around this concept, different *ICA* algorithms have been developed: FastICA (Aapo Hyvärinen & Oja, 1997), *JADE* (Cardoso & Souloumiac, 1993; Cardoso Resources, 2016), InfoMax (Bell & Sejnowski, 1995), Mutual Information Least Dependent Component Analysis (Stögbauer, Kraskov, Astakhov, & Grassberger, 2004), Stochastic Non-Negative Independent Components Analysis (Astakhov, Stögbauer, Kraskov, & Grassberger, 2006), *RADICAL* (Learned-Miller & Fisher III, 2003), etc.

**2.2 Independence and linear correlation**

Suppose a random variable  $X$  that can take  $v$  different values, with the probability that  $X = x_i$  defined to be  $P(X = x_i) = p_i$ . Then the expectation ( $E$ ) of  $X$  is defined as

$$E(X) = x_1 p_1 + x_2 p_2 + \dots + x_v p_v \quad (1)$$

and its variance ( $Var$ )

$$Var(X) = E(X^2) - (E(X))^2 \quad (2)$$

Suppose that  $X$  and  $Y$  are two random variables with expected values  $E(X)$ ,  $E(Y)$  and variances  $Var(X)$ ,  $Var(Y)$ , respectively. The covariance ( $Cov$ ) of  $(X, Y)$  is defined by

$$Cov(X, Y) = E(XY) - E(X)E(Y) \quad (3)$$

and its correlation ( $Corr$ )

$$Corr(X, Y) = \frac{Cov(X, Y)}{(Var(X)Var(Y))^{\frac{1}{2}}} \quad (4)$$

The  $Corr(X, Y)$  measures the linear dependence between two variables  $X$  and  $Y$ , giving a value between -1 and +1 both inclusive.

Two random variables  $X$  and  $Y$  are uncorrelated when their correlation coefficient is zero:  $Corr(X, Y) = 0$ . Being uncorrelated is the same as having zero covariance and therefore, from (3):

$$E(XY) = E(X)E(Y) \quad (5)$$

If  $X$  and  $Y$  are independent, then they are uncorrelated and consequently  $Corr(X, Y) = 0$ . If  $Corr(X, Y) \neq 0$ , then  $X$  and  $Y$  present some grade of dependence. If  $Corr(X, Y) = 0$ , then  $X$  and  $Y$  can be either dependent or independent.

Suppose that  $(x_m, y_m)$  with  $m = 1, 2, \dots, v$  form a sample from a pair of random variables  $X$  and  $Y$ . The distance covariance ( $dCov$ ) (Székely, Rizzo, & Bakirov, 2007) is defined as:

$$dCov^2(X, Y) = \frac{1}{v^2} \sum_{j, h=1}^v A_{j, h} B_{j, h} \quad (6)$$

where

$$\begin{aligned} A_{j,h} &:= a_{j,h} - \bar{a}_j - \bar{a}_h + \bar{a}_{..}, & a_{j,h} &= \|x_j - x_h\| & j, h &= 1, 2, \dots, v \\ B_{j,h} &:= b_{j,h} - \bar{b}_j - \bar{b}_h + \bar{b}_{..}, & b_{j,h} &= \|y_j - y_h\| & j, h &= 1, 2, \dots, v \\ \bar{a}_j &= \frac{1}{v} \sum_{h=1}^v a_{jh}, & \bar{a}_h &= \frac{1}{v} \sum_{j=1}^v a_{jh}, & \bar{a}_{..} &= \frac{1}{v^2} \sum_{j,h=1}^v a_{jh} \end{aligned}$$

The notation is similar for  $\bar{b}_j, \bar{b}_h$  and  $\bar{b}_{..}$ .

The distance variance ( $dVar$ ) of  $X$  is

$$dVar^2(X) = dCov^2(X, X) = \frac{1}{v^2} \sum_{j,h=1}^v A_{j,h}^2 \quad (7)$$

The distance correlation ( $dCorr$ ) (Székely, Rizzo, & Bakirov, 2007) of two random variables  $X$  and  $Y$  is obtained by dividing their distance covariance by the product of their distance standard deviations. The distance correlation is

$$dCorr^2(X, Y) = \frac{dCov^2(X, Y)}{(dVar^2(X) dVar^2(Y))^{\frac{1}{2}}} \quad (8)$$

The distance correlation is a measure of the statistical dependence between two random variables  $X$  and  $Y$ . This measure of dependence is zero if and only if  $X$  and  $Y$  are independent.  $Corr(X, Y) = 0$  does not imply independence while  $dCorr(X, Y) = 0$  implies independence.

### 3. The LCC algorithm

In this section, the proposed Linear Correlation between Components (LCC) algorithm is presented. This algorithm addresses the problem of automatically finding the optimal number of  $IC$ 's in a simple and fast way. It makes use of: (i) the *JADE* algorithm to calculate the  $NIC$ , (ii) the linear correlation between components, and (iii) the fact that a decomposed  $IC$  appears when the  $NIC$  is superior to  $k$ .

Given a set of  $N$  mixed signals and a  $NIC$ , the *JADE* algorithm provides a model of  $NIC$  components. If  $NIC$  varies between 1 and a specific number  $MNIC$  where  $1 \leq k < MNIC < N$ , *JADE* provides  $MNIC$  different models.

Consider the  $MNIC$  models to be divided into two groups in the following way: The first group contains the models with a number of  $IC$ 's less than or equal to  $k$ . The second group contains the models with a number of components



superior to  $k$ . For each model belonging to the first group the components are independent and therefore, any pair of different components will have a correlation equal to zero. The maximum and optimal number of independent components is located in the model  $k$ .

In the model  $k+1$ , *JADE* generates  $k+1$  components. Among these components, it is not possible to get more than  $k$  independent components as the maximum is  $k$ . It means that, at least, one component is not completely independent from the other components.

Thus, in the model  $k+1$  two components of the *JADE* output result from a decomposition of an *IC* in the model  $k$  into two components. These two components are obviously not independent as they are dependent on one *IC*. From a quantitative point of view, the absolute value of the correlation between both components will be greater than zero.

Keeping this in mind, *LCC* calculates the correlation between all pairs of different components for each model using (4). These calculations stop when a model  $k+1$  where two different components have an absolute correlation value different from zero is found.

In the model  $k$ , the optimal number of *IC*'s corresponds with the maximum number of *IC*'s where the correlation between all different pairs of components is zero.

The proposed algorithm is described as follows:

Algorithm *LCC*. Linear Correlations between Components.

**Input:** A set of  $N$  mixed signals.

**Output:** Optimal number  $k$  and its *IC*'s.

**for each model of  $NIC \geq 2$  components repeat**

1. Calculate the *NIC* independent components using *JADE*
2. Calculate for each pair of components  $(a, b)$  the  $Corr(a, b)$  with  $1 \leq a < b \leq NIC$

**until for a particular model  $k+1$ , two different components  $(p, m)$  have  $|Corr(p, m)| > 0.1$**

3. Select the  $k$  and its *IC*'s

**Some remarks about the algorithm:** (1) symbols  $a$  and  $b$  are integers to identify a pair of components in a model; (2) symbols  $p$  and  $m$  are integers to identify the first pair of components detected where  $|Corr(p, m)| > 0.1$ ; (3) the absolute value of the correlation  $|Corr(p, m)|$  is lower bounded by 0.1 instead of 0 to avoid small quantities near zero that could stop the algorithm; (4) *JADE* needs

that  $N > k$  in order to obtain the optimal number of IC's; and (5) the LCC algorithm starts with a model of  $NIC=2$ .

#### 4. Testing the algorithm

In this section, the LCC algorithm is tested using synthetic data. First a set of mixed signals from  $N_p$  pure signals is generated. Then, the LCC algorithm uses these mixed signals as input to find  $k = N_p$  and its IC's (similar to the  $N_p$  pure signals).

This section also presents some tests using the PCA method, the Durbin-Watson criterion and the *RV\_ICA\_by\_Blocks* method mentioned in the introduction. These tests allow showing some differences with the LCC method.

Fig. 1 shows a set of six pure signals with different levels of noise.

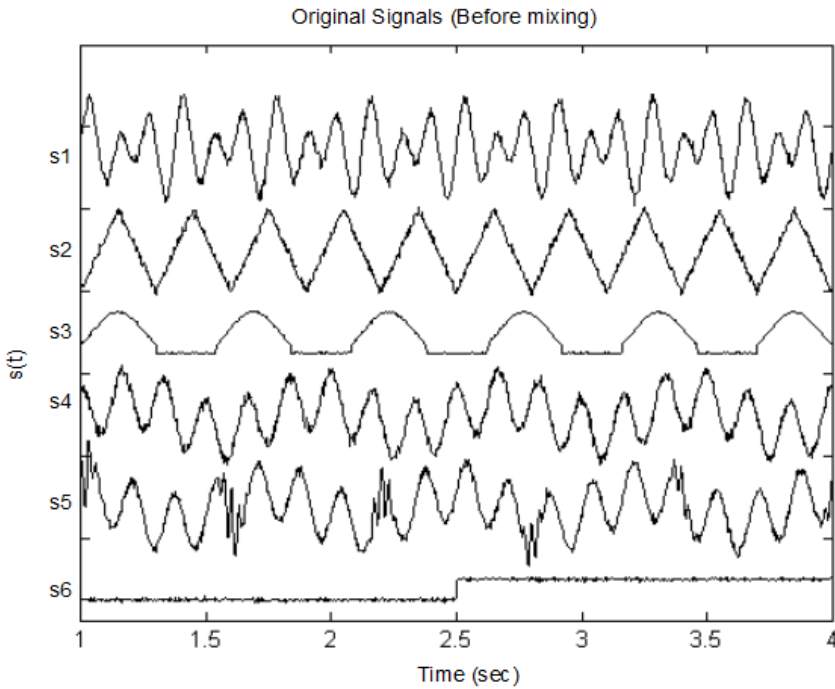


Figure 1: Six pure signals before mixing.

Table 1 shows the distance correlations between all different pairs of signals calculated using the formula (8).

**Table 1**  
Distance correlation between the 6 pure signals  $s_1, s_2, s_3, s_4, s_5, s_6$

$dCorr(s_i, s_j)$	$s_2$	$s_3$	$s_4$	$s_5$	$s_6$
$s_1$	0.0767	0.0399	0.0725	0.0791	0.0186
$s_2$		0.0758	0.0971	0.0578	0.0151
$s_3$			0.0643	0.0987	0.0101
$s_4$				0.0188	0
$s_5$					0.0254

The  $dCorr$  values found in Table 1 are close to zero. It means that there are small dependences between all different pairs of signals. However, these small values are not a problem for the performance of *JADE* and *LCC*. We can consider that the six signals  $s_1, s_2, s_3, s_4, s_5, s_6$  form a set of pure signals.

#### 4.1 Test using 3 pure signals and *LCC*

Fig. 2 shows a set of six mixed signals  $M (= M_1, M_2, M_3, M_4, M_5, M_6)$  using  $N_p = 3$  pure signals  $S (= s_1, s_2, s_3)$ . These signals are constructed by using  $M = A \times S$  where  $A$  is the following mixing matrix of random numbers between -1 and 1:

$$A = \begin{bmatrix} -0.23 & 0.51 & 0.19 \\ 0.58 & 0.09 & -0.99 \\ -0.24 & 0.71 & -0.44 \\ 0.13 & -0.40 & -0.31 \\ -0.76 & -0.15 & 0.91 \\ -0.19 & -0.17 & -0.69 \end{bmatrix}$$

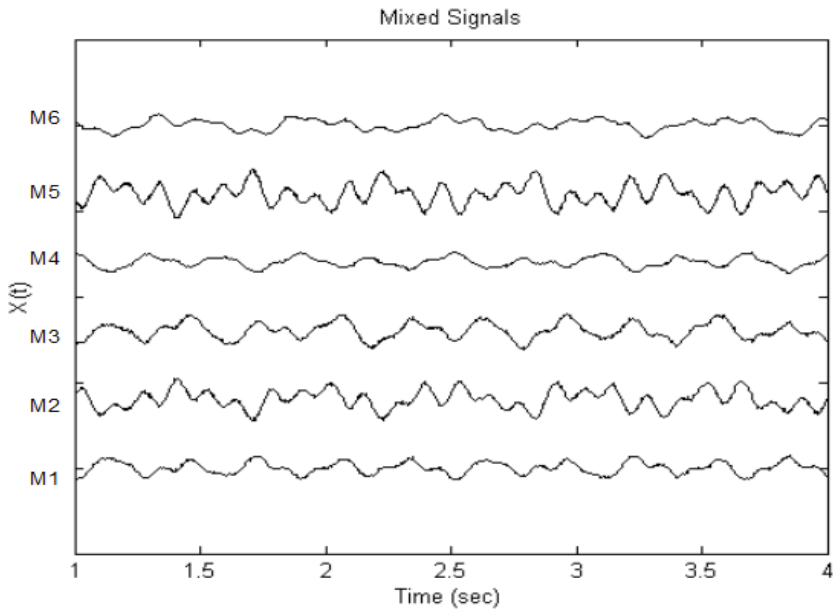


Figure 2: Six mixed signals.

The *LCC* finds  $k=3$  and Fig. 3 shows the *IC's* of model 3.

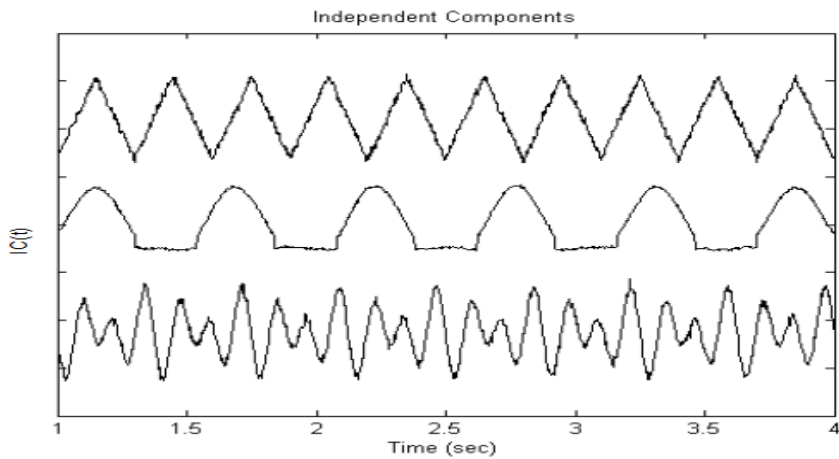


Figure 3: The three optimal *IC's* of model 3.

Fig. 4 shows the components (2, 3, 4) of model 4. The components (2, 4) are related to the IC placed at the bottom of Fig. 3 and they are dependent. Their  $|Corr(2, 4)|=0.63$ . This value could stop LCC but in this case, this algorithm stops with the pair (2,3). Their  $|Corr(2, 3)|=0.43$ . This correlation does not need to be the maximum because other pairs of components can also be dependent.

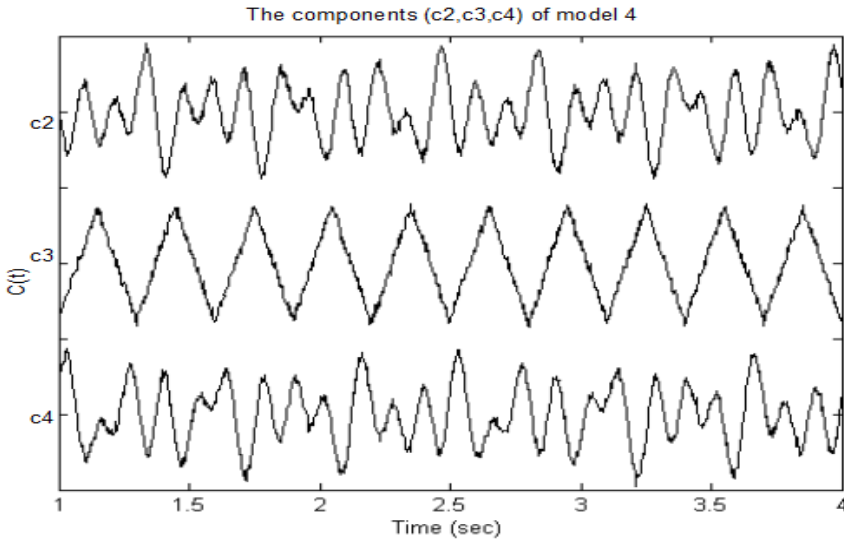


Figure 4: Components (c2, c3, c4) of model 4.

Table 2 summarizes this test using 3 pure signals.

**Table 2**  
**Optimal number  $k=3$  of IC's for the test using 3 pure signals**

$S$	$N_p$	$N$	$k+1$	$ Corr(p, m) $ in model $k+1$	$MC=\max( Corr(p, m) )$ in model $k+1$	$k$
$(s1, s2, s3)$	3	6	4	$ Corr(2, 3) =0.43$	$MC= Corr(2, 4) =0.63$	3

#### 4.2 Additional tests using LCC

Table 3 shows the results of some tests using other sets of pure signals  $S$ . The  $k$  value for all sets is correct.

**Table 3**  
**Optimal number  $k$  of IC's for some tests using different sets of pure signals.**

$S$	$N_p$	$N$	$k+1$	$ Corr(p, m) $ in model $k+1$	$MC=\max( Corr(p, m) )$ in model $k+1$	$k$
$(s_1, s_2, s_3, s_4)$	4	10	5	$ Corr(1, 2) =0.34$	$MC= Corr(2, 3) =0.57$	4
$(s_1, s_2, s_3, s_4, s_5)$	5	10	6	$ Corr(5, 6) =1$	$MC= Corr(5, 6) =1$	5
$(s_1, s_2, s_3, s_4, s_5, s_6)$	6	10	7	$ Corr(1, 7) =0.23$	$MC= Corr(5, 7) =0.74$	6

The  $MC$  values found in Table 2 and 3 show that, in the model  $k+1$ , there may be more than one pair able to stop the  $LCC$  algorithm. The pair of components chosen to stop  $LCC$  depends on the order that  $LCC$  analyses the pairs.

It is also worth to note that the  $Corr$  values do not have to decrease with the number of pure signals.

#### **4.3 Additional tests using the $PCA$ method, the Durbin-Watson criterion and the $RV\_ICA\_by\_Blocks$**

In this subsection, some alternative methods to determine  $k$  (the  $PCA$  method, the Durbin-Watson criterion and the  $RV\_ICA\_by\_Blocks$ ) are tested using four sets of mixed signals:

- A) 10 mixed signals generated from the combination of 3 pure signals ( $s_1, s_2, s_3$ ).
- B) 10 mixed signals generated from the combination of 4 pure signals ( $s_1, s_2, s_3, s_4$ ).
- C) 10 mixed signals generated from the combination of 5 pure signals ( $s_1, s_2, s_3, s_4, s_5$ ).
- D) 10 mixed signals generated from the combination of 6 pure signals ( $s_1, s_2, s_3, s_4, s_5, s_6$ ).

**PCA method:**

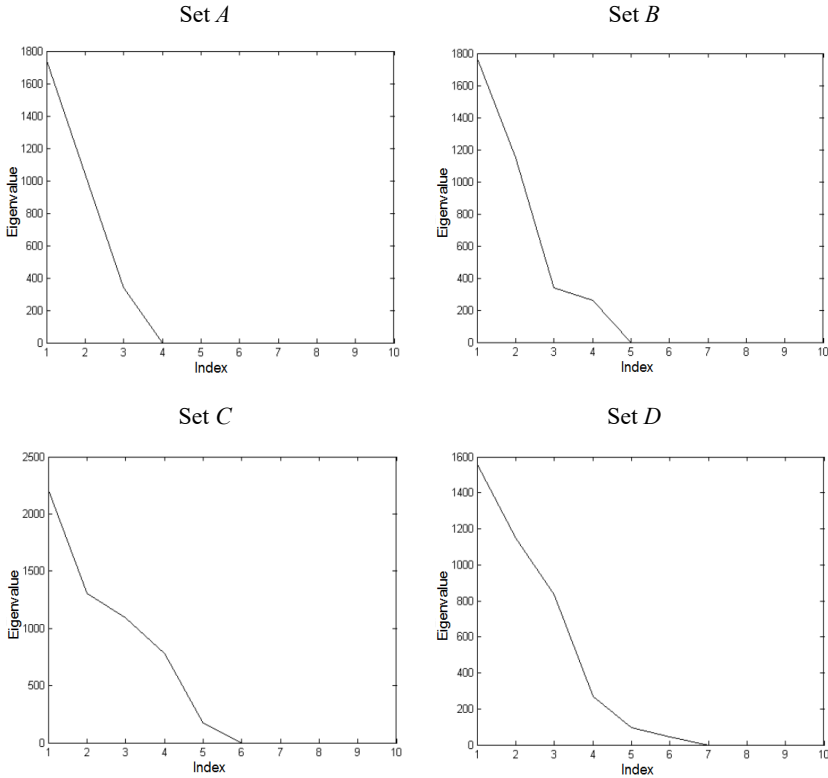


Fig. 5: Scree plots (Sets A, B, C and D) of the PCA method used to determine  $k$ . The *Index* represents the order of the eigenvalues.

The *ACP* method to determine  $k$  is explained in (Semmlow, 2004). Fig. 5 shows four scree plots for the sets A, B, C and D.  $k$  is the number where a shape break is located. For the set A,  $k=3$ ; for the set B,  $k=3$  (instead of 4); for the set C,  $k=5$ ; and for the set D, it seems that  $k=5$  (instead of 6).

**Durbin-Watson criterion:**

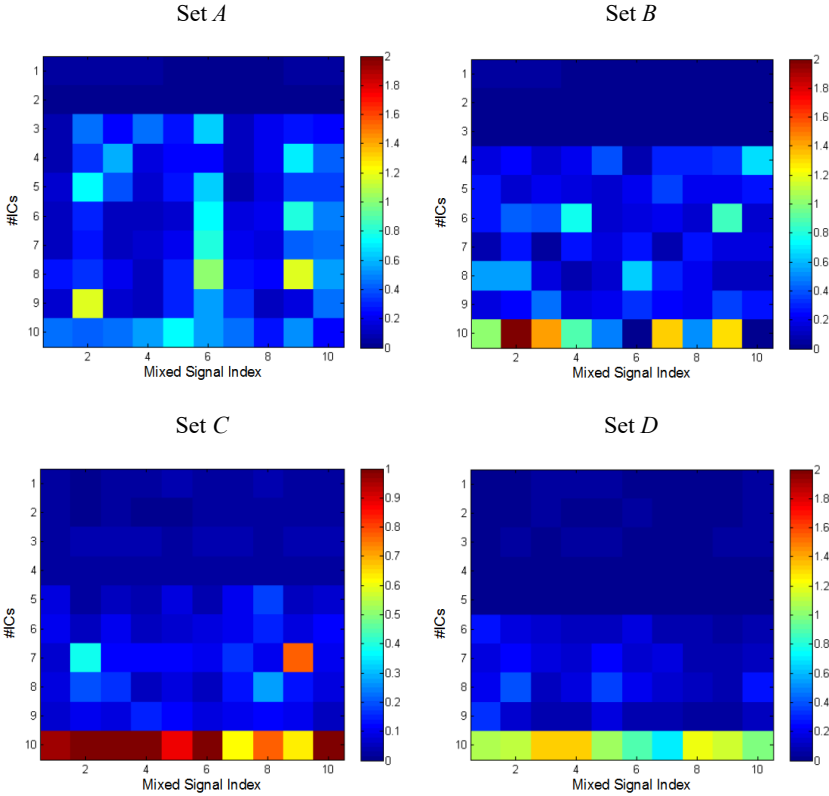


Fig. 6: Colour plots (Sets  $A$ ,  $B$ ,  $C$  and  $D$ ) of the Durbin-Watson criterion used to determine  $k$ .

The method to determine  $k$  using the Durbin-Watson criterion is explained in detail in (Bouveresse, & Rutledge, 2016). This criterion estimates the signal-to-noise ratio of a structured signal. It cannot be used in the case of unstructured signals. Its value is close to 0 when there is not much noise. This method is applied to a set of models (from 1 to  $maxICs$ ). For the sets  $A$ ,  $B$ ,  $C$  and  $D$ ,  $maxICs=10$ .

Fig. 6 shows four colour plots of the Durbin-Watson values for the sets  $A$ ,  $B$ ,  $C$  and  $D$ .  $k$  corresponds with the highest  $IC$ -row among the rows that have small Durbin-Watson values (it means a high signal-to-noise ratio). For the set  $A$ ,  $k=2$  (instead of 3); for the set  $B$ ,  $k=3$  (instead of 4); for the set  $C$ ,  $k=4$  (instead of 5); and for the set  $D$ ,  $k=5$  (instead of 6).



***RV\_ICA\_by\_Blocks:***

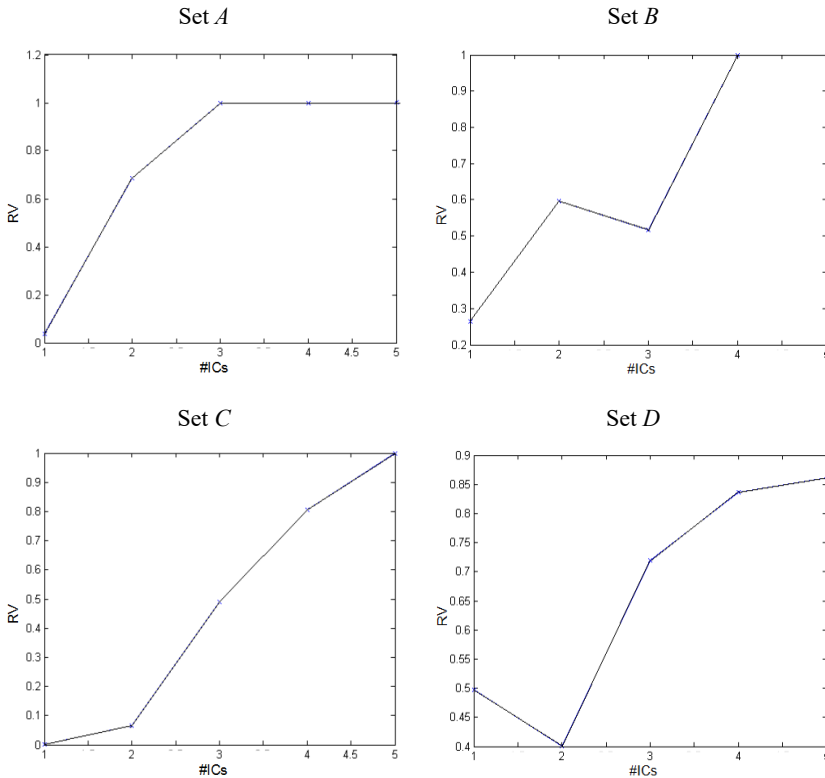


Fig. 7: Plots (Sets A , B, C and D) of the *ICA\_by\_Blocks* method to determine *k*.

The *RV* coefficient (Escoufier, 1973; Bouveresse, & Rutledge, 2016) measures the similarity of information between two matrices. It ranges from 0 to 1. When *RV* is equal to 1, it means that their information content is similar.

In the *RV\_ICA\_by\_Blocks* method, the data are split into two representative subsets, and several *ICA* models (from 1 to *maxICs* *IC*'s) are calculated. For each number of *IC*'s tested, a *RV* coefficient is calculated between the *S* matrices (see Subsection 2.1) obtained for the two data sets.

Fig. 7 shows some plots of the *ICA\_by\_Blocks* method for the sets *A*, *B*, *C* and *D*. The *maxICs* value cannot exceed half the number of mixed signals. In this case, *maxICs*=5. *k* corresponds with the first *IC* where *RV*=1. For the set *A*, *k*=3; for the set *B*, *k*=4; for the set *C*, *k*=5; and for the set *D*, it is not possible to determine *k* because the number of mixed signals is not enough.

## 5. Discussion

*JADE* finds *IC*'s for all models with  $NIC \leq k$ . For the models with  $NIC > k$ , *JADE* cannot create more than  $k$  *IC*'s because this number is the maximum possible. However, *JADE* can generate additional components by decomposing an *IC*. For example, in the model  $k+1$ , two components are created from one *IC* of model  $k$ . Both components will have some grade of dependence and their correlation will be different from zero.

The calculation of the correlation between pairs of different components allows finding the model  $k+1$  and consequently, the  $k$  number of *IC*'s is clearly identified.

Unlike the mentioned methods in Section 1 (and Subsection 4.3), the proposed unsupervised *LCC* method does not need either previous knowledge or preparing the data (as for example creating blocks). This method gives automatically  $k$  and there is no need to analyse a graphic representation. It only checks the linear correlation between components from the different models obtained with *JADE*.

The considerations for setting the threshold value in *LCC* are the following. In practice, the *Corr* values between pairs of *IC*'s for each model (from 2 to  $k$ ) are very close to zero. The *Corr* values in the model  $k+1$  that allow stopping the *LCC* algorithm are significantly higher than 0.1. Consequently, a threshold in *LCC* of 0.1 is a good choice because it is far from both values.

In theory, the *LCC* algorithm could be modified to use *dCorr* instead of *Corr*. In general, the values of *dCorr* between pairs of *IC*'s for each model (from 2 to  $k$ ) are more variable than the *Corr* values and they can be close to 0.1. Consequently, the threshold in *LCC* should be higher than 0.1. However, *dCorr* is not necessary because *Corr* offers a good behavior, it is simpler and its threshold of 0.1 is well defined.

Our experiments have shown a good performance of the proposed algorithm.

Subsection 4.3 shows different tests to determine  $k$  using the *PCA* method, the Durbin-Watson criterion and the *RV\_ICA\_by\_Blocks*.

The *ACP* method is simple but not very precise. However, it has correctly determined  $k$  for the sets  $A$  and  $C$ .

By contrast, the Durbin-Watson criterion, with the help of a colour plot, can determine  $k$  more accurately. In general, this criterion can only be used in structured signals. The  $k$  obtained, for any set of mixed signals  $A$ ,  $B$ ,  $C$  and  $D$ , has always been equal to the number of pure signals minus one.

The *RV\_ICA\_by\_Blocks* method has found the right  $k$  for the sets  $A$ ,  $B$ , and  $C$  but it has been unable to determine  $k$  for set  $D$ .

In general, a “*Blocks*” method needs to select representative and comparable data blocks (For example, two blocks). Its result is highly dependent on the distribution of the mixed signals between the blocks. Normally, the “*Blocks*” algorithm is repeated a number of times on random distributions of the mixed signals into the two blocks. Then, a graphical representation of all the results

shows whether repeatable results can be obtained or not. Furthermore, as noted with set  $D$ , the “Blocks” methods cannot find  $k$  when the number of mixed signals is less than  $2 \times k$ .

## 6. Conclusion

A simple, effective and fast new method, *LCC*, to find the optimal number of independent components  $k$  from an *ICA* has been presented. It is an unsupervised method that obtains  $k$  automatically.

The *LCC* finds  $k$  by checking linear correlation between components of the different models obtained with *JADE*.

## Acknowledgements

This work was supported in part by the *UNED* project GID2016-6-1, the Spanish Ministry of Economy and Competitiveness under Projects DPI2014-55932-C2-2-R and ENE2015-64914-C3-2-R and FEDER funds.

## References

- Astakhov, S. A., Stögbauer, H., Kraskov, A., & Grassberger, P. (2006). Monte Carlo algorithm for least dependent non-negative mixture decomposition. *Analytical Chemistry*, 78(5), 1620–1627. <http://doi.org/10.1021/ac051707c>
- Bell, A. J., & Sejnowski, T. J. (1995). An information-maximization approach to blind separation and blind deconvolution. *Neural Computation*, 7(6), 1129–1159. <http://doi.org/10.1162/neco.1995.7.6.1129>
- Bouveresse, J-R., Rutledge D.N (2016). Independent Components Analysis: Theory and Application. *Resolving Spectral Mixtures: with applications from ultrafast time-resolved spectroscopy to super-resolution imaging* (pp. 225–273). Lille: Elsevier
- Cardoso, J. F., & Souloumiac, A. (1993). Blind beamforming for non-gaussian signals. *IEE Proceedings F Radar and Signal Processing*. <http://doi.org/10.1049/ip-f-2.1993.0054>
- Cardoso J-F. Cardoso Resources. Accessed 30/08/2016. <http://perso.telecom-paristech.fr/~cardoso/Algo/Jade/jadeR.m>
- Escoufier, Y. (1973). Le traitement des variables vectorielles. *Biometrics*, 29, 751–760.
- Hao, J., Zou, X., Wilson, M. P., Davies, N. P., Sun, Y., Peet, A. C., & Arvanitis, T. N. (2009). A comparative study of feature extraction and blind source separation of independent component analysis (ICA) on childhood brain

- tumour 1H magnetic resonance spectra. *NMR in Biomedicine*, 22(8), 809–818. <http://doi.org/10.1002/nbm.1393>
- He, T., Clifford, G., & Tarassenko, L. (2006). Application of independent component analysis in removing artefacts from the electrocardiogram. *Neural Computing and Applications*, 15(2), 105–116. <http://doi.org/10.1007/s00521-005-0013-y>
- Hyvärinen, A., & Oja, E. (1997). A Fast Fixed-Point Algorithm for Independent Component Analysis. *Neural Computation*, 9(7), 1483–1492. <http://doi.org/10.1162/neco.1997.9.7.1483>
- Hyvärinen, A., & Oja, E. (2000). Independent component analysis: algorithms and applications. *Neural Networks: The Official Journal of the International Neural Network Society*, 13(4-5), 411–30. [http://doi.org/10.1016/S0893-6080\(00\)00026-5](http://doi.org/10.1016/S0893-6080(00)00026-5)
- Jolliffe, I. T. (2002). Principal Component Analysis, Second Edition. *Encyclopedia of Statistics in Behavioral Science*, 30(3), 487. <http://doi.org/10.2307/1270093>
- Jouan-Rimbaud Bouveresse, D., Moya-González, A., Ammari, F., & Rutledge, D. N. (2012). Two novel methods for the determination of the number of components in independent components analysis models. *Chemometrics and Intelligent Laboratory Systems*, 112, 24–32. <http://doi.org/10.1016/j.chemolab.2011.12.005>
- Krishnaveni, V., Jayaraman, S., Manoj Kumar, P. M., Shivakumar, K., & Ramadoss, K. (2005). Comparison of Independent Component Analysis Algorithms for Removal of Ocular Artifacts from Electroencephalogram. *Measurement Science Review*, 5.
- Learned-Miller, E. G., & Fisher III, J. W. (2003). ICA Using Spacings Estimates of Entropy. *The Journal of Machine Learning Research*, 4, 1271–1295. <http://doi.org/10.1162/jmlr.2003.4.7-8.1271>
- Rutledge, D. N., & Barros, A. S. (2002). Durbin-Watson statistic as a morphological estimator of information content. *Analytica Chimica Acta*, 454(2), 277–295. [http://doi.org/10.1016/S0003-2670\(01\)01555-0](http://doi.org/10.1016/S0003-2670(01)01555-0)
- Rutledge, D. N., & Jouan-Rimbaud Bouveresse, D. (2013). Independent Components Analysis with the JADE algorithm. *TrAC - Trends in Analytical Chemistry*. <http://doi.org/10.1016/j.trac.2013.03.013>
- Semmlow, J. (2004). Biosignal and biomedical image processing: MATLAB-based applications. *CRC Press*. <http://doi.org/10.1201/9780203024058>

- Stögbauer, H., Kraskov, A., Astakhov, S. A., & Grassberger, P. (2004). Least-dependent-component analysis based on mutual information. *Physical Review E*, 70, 1–17. <http://doi.org/10.1103/PhysRevE.70.066123>
- Székely, G. J., Rizzo, M. L., & Bakirov, N. K. (2007). Measuring and testing dependence by correlation of distances. *Annals of Statistics*, 35(6), 2769–2794. <http://doi.org/10.1214/009053607000000505>
- Wang, G., Ding, Q., & Hou, Z. (2008). Independent component analysis and its applications in signal processing for analytical chemistry. *TrAC - Trends in Analytical Chemistry*, 27(4), 368–376. <http://doi.org/10.1016/j.trac.2008.01.009>



## CAPÍTULO 8

---

### CONCLUSIONES

---

En este capítulo se exponen las conclusiones de esta tesis agrupadas en los diferentes artículos compendiados.

#### **ART. 1:**

- *UMED* es un método no supervisado que permite detectar y caracterizar los eventos de una señal cuando no se tiene ningún conocimiento sobre su naturaleza.
- *UMED* proporciona una solución óptima para la detección de eventos con la ayuda del algoritmo de Clustering *HC* y un índice de validación interna  $S_{dbw}$ .
- *UMED* utiliza una ventana óptima para captar los coeficientes *AR* a lo largo de la señal.
- *UMED1* y *UMED2* son dos métodos para extender el uso de *UMED* a señales de larga duración y poder caracterizar (o identificar) los eventos en tiempo real.
- *UMED* permite detectar los artefactos en una señal *EEG* con el objetivo de estudiarlos para realizar estudios psicofisiológicos o bien suprimirlos.

## Capítulo 8: Conclusiones

- *UMED* se adapta perfectamente al estudio de eventos en señales multicanal y proporciona resultados invariantes al sistema de adquisición utilizado para cada canal.
- *UMED* es un método no supervisado y no necesita ningún proceso de aprendizaje para la detección de eventos.
- El índice de validación *CCC* del algoritmo *HC* permite acelerar el procesado de *UMED*.

### **ART. 2:**

- El método *UMEDC* permite clasificar de forma no supervisada un conjunto de señales en el que la localización de los eventos y la duración de los estados en cada señal son variables.
- *UMEDC* clasifica las señales según su comportamiento. La duración de cada señal puede variar.
- *UMEDC* transforma las señales en secuencias temporales con la ayuda de *UMED* y, posteriormente, clasifica las secuencias utilizando las distancias de todos los pares de secuencias.
- *UMEDC* también clasifica las señales transformando las secuencias temporales en vectores de características con el mismo número de componentes.



## Capítulo 8: Conclusiones

- Se ha desarrollado un método basado en *SC* para transformar un conjunto de secuencias de diferente longitud en un conjunto de vectores de características de igual longitud. También es posible encontrar la señal representativa de cada grupo.
- *UMEDC* con la ayuda de los coeficientes *AR* proporciona una solución independiente del proceso de adquisición de cada señal.
- Cuando el número de señales a clasificar es pequeño *UMED* obtiene la ventana óptima utilizando el valor óptimo de *S\_dbw*. Cuando el número de señales a clasificar es grande, la detección de eventos de cada señal puede realizarse utilizando la ventana en la que el *CCC* es máximo. Con esta selección, *UMEDC* puede procesar todas las señales rápidamente.

### **ART. 3:**

- El método *GS* encuentra el *NO* de un conjunto de objetos utilizando las nuevas representaciones de los objetos obtenidas con *SC*.
- *GS* combina el índice de validación interna Silhouette con el concepto de “Local Scaling”.

## Capítulo 8: Conclusiones

- *GS* es robusto respecto al ruido y puede encontrar los clusters de objetos distribuidos en diferentes densidades.
- *GS* puede adaptarse para trabajar con un número importante de objetos mediante *GSWB*. El algoritmo *GSWB* puede encontrar *NO* y un contenido de los clusters similar al obtenido con el algoritmo de Nyström.
- *GS* se ha probado con datos sintéticos y *GSWB* con imágenes. Ambos han podido evaluarse mediante inspección visual. El índice de Silhouette también puede utilizarse para comparar cuantitativamente el resultado de una segmentación de una imagen utilizando *GSWB* y Nyström\_ *WB*.
- La combinación *SC+GS* es útil para clasificar objetos. Sin embargo no se utiliza en *UMED* ya que la combinación *HC+S\_dbw* además de obtener buenos resultados puede hacer uso del coeficiente *CCC* para acelerar el tiempo de procesado.

### **ART. 4:**

- El método *LCC* determina el número óptimo de componentes independientes de una señal *MC*, registrada con el mismo tipo de sensores en todos los canales, de forma automática y no supervisada.

## *Capítulo 8: Conclusiones*

- *LCC* encuentra el número óptimo de componentes independientes analizando la correlación lineal entre componentes de los diferentes modelos obtenidos con *JADE*.
- *LCC* no necesita para determinar el óptimo número de componentes independientes una representación gráfica, ni información previa sobre las componentes, ni agrupar los canales en grupos. Tampoco tiene limitaciones respecto al tipo de componentes independientes a encontrar.
- *LCC* necesita que el número de canales de la señal *MC* sea mayor que el número óptimo de componentes independientes. También está limitado por *JADE* cuando es necesario calcular un número alto de componentes.

## Bibliografía

- [Aka74] Akaike, H. (1974). A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6):716-723.
- [AL89] Auger, I. E. and Lawrence, C. E. (1989). Algorithms for the optimal identification of segment neighborhoods. *Bulletin of Mathematical Biology*, 51(1):39-54.
- [BN93] Basseville, M., & Nikiforov, I. V. (1993). Detection of Abrupt Changes: Theory and Application. *Change*, 2(4), 729–730.
- [BS95] Bell, A. J., & Sejnowski, T. J. (1995). An information-maximization approach to blind separation and blind deconvolution. *Neural Computation*, 7(6), 1129–1159.
- [BRS+07] Benítez, M. C., Ramírez, J., Segura, J. C., Ibáñez, J. M., Almendros, J., García-Yeguas, A., & Cortés, G. (2007). Continuous HMM-based seismic-event classification at deception Island, Antarctica. In *IEEE Transactions on Geoscience and Remote Sensing* (Vol. 45, pp. 138–146).
- [Bou09] Boussemart, Yves et al. (2009). Comparing Learning Techniques for Hidden Markov Models of Human Supervisory Control Behavior. *AIAA Aerospace Conference*, 6-9.
- [CS93] Cardoso, J. F., & Souloumiac, A. (1993). Blind beamforming for non-gaussian signals. *IEE Proceedings F Radar and Signal Processing*.
- [CG00] Chen, J. and Gupta, A. K. (2000). Parametric statistical change point analysis. Birkhauser.
- [CMY99] Chou, Y. M., Mason, R. L., & Young, J. C. (1999). Power comparisons for a hotelling's T2 statistic. *Communications in Statistics Part B: Simulation and Computation*.

- [DB79] Davies, D. L., & Bouldin, D. W. (1979). A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1(2), 224–227.
- [DHY95] Davis, R., Huang, D., and Yao, Y.C. (1995). Testing for a change in the parameter values and order of an autoregressive model. *Annals of Statistics*, 23(11), 282–304.
- [DLR06] Davis, R. A., Lee, T. C. M., and Rodriguez-Yam, G. A. (2006). Structural break estimation for nonstationary time series models. *JASA*, 101(473), 223-239.
- [DLR77] Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1), 1–38.
- [DTG+14] Dimitrios D., Trajcevski G., Gunopulos D., Aggarwal C. (2014). Chapter 15: Time-series Data Clustering, *Data Clustering Algorithms and Applications. Chapman & Hall/CRC Data Mining and Knowledge Discovery Series, Vipin Kumar, University of Minnesota*
- [Eve79] Everitt, B. S. (1979). A Monte Carlo Investigation of the Robustness of Hotelling’s One- and Two-Sample T<sub>2</sub> Tests. *Journal of the American Statistical Association*, 74(365), 48–51.
- [FM83] Fowlkes, E. B., & Mallows, C. L. (1983). A Method for Comparing Two Hierarchical Clusterings. *Journal of the American Statistical Association*, 78(383), 553.
- [Gom08] Gombay, E. (2008). Change detection in autoregressive time series. *J. Multivariate Anal.*, 99(3), 451 – 464.
- [GGR+02] Ganti, Gehrke J., Ramakrishnan R., Loh W. (2002). A framework for measuring changes in data

- characteristics. *Journal of Computer and System Sciences*, 64, 542–578.
- [HV01] Halkidi, M., & Vazirgiannis, M. (2001). Clustering Validity Assessment : Finding the optimal partitioning of a data set. In *In Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on* (pp. 187–194).
- [HVB00] Halkidi, M., Vazirgiannis, M., & Batistakis, Y. (2000). Quality scheme assessment in the clustering process. *Principles of Data Mining and Knowledge Discovery*, 265–276.
- [HME16] Hamed M. S., Mansour M., Elrazik E. (2016). MCUSUM control chart procedure: monitoring the process mean with application. *Journal of Statistics: Advances in Theory and Applications*. 16(1), 105-132.
- [Ho05] Ho, S. (2005). A martingale framework for concept change detection in time-varying data streams. *Proceedings of the 22nd International Conference on Machine Learning (ICML-05)*, (2004), 321–327.
- [HPS07a] Huskova, M., Praskova, Z., and Steinebach, J. (2007a). On the detection of changes in autoregressive time series, i. Asymptotics. *Journal of Statistical Planning and Inference*, 137(4), 1243–1259.
- [HPS07b] Huskova, M., Praskova, Z., and Steinebach, J. (2007b). On the detection of changes in autoregressive time series, ii. Resampling procedures. *Journal of Statistical Planning and Inference*, 138(6), 1697–1721.
- [HZ14] Hui Xiong, Zhongmou Li. (2014). Chapter 23: Clustering Validation Measure, Data Clustering Algorithms and Applications. *Chapman & Hall/CRC Data Mining and Knowledge Discovery Series*, Vipin Kumar, University of Minnesota.

- [HO97] Hyvärinen, A., & Oja, E. (1997). A Fast Fixed-Point Algorithm for Independent Component Analysis. *Neural Computation*, 9(7), 1483–1492.
- [Jai10] Jain, A. K. (2010). Data clustering: 50 years beyond K-means. *Pattern Recognition Letters*, 31(8), 651–666.
- [JD88] Jain, A. K., & Dubes, R. C. (1988). Algorithms for Clustering Data. *Prentice Hall*.
- [JMF99] Jain, A. K., Murty, M. N., & Flynn, P. J. (1999). Data clustering: A review. *ACM Computing Surveys*, 31(3), 264–323.
- [KGP01] Kalpakis K., Gada D., Puttagunta V. (2001). Distance measures for effective clustering of ARIMA time-series. *ICDM*, 273–280, 2001.
- [KR05] Kaufman, L., & Rousseeuw, P. J. (2005). *Finding Groups in Ordinal Data. An Introduction to Cluster Analysis*. Wiley-Interscience.
- [KBG04] Kifer D., Ben-David S., Gehrke J.. Detecting changes in data streams. In *VLDB*, 2004.
- [KFE12] Killick, R., Fearnhead, P., & Eckley, I. a. (2012). Optimal detection of changepoints with a linear computational cost. *Journal of the American Statistical Association*, 107(500), 1590–1598.
- [KR05] Kim, M., & Ramakrishna, R. S. (2005). New indices for cluster validity assessment. *Pattern Recognition Letters*, 26(15), 2353–2363.
- [KMP+11] Kraus, J. M., Müssel, C., Palm, G., & Kestler, H. A. (2011). Multi-objective selection for collecting cluster alternatives. *Computational Statistics*, 26(2), 341–353.
- [LHM+12] Lawhern V., Hairston WD, McDowell K., Westerfield M., K. R. (2012). Detection and classification of subject-generated artifacts in EEG signals using

- autoregressive models. *Neuroscience Methods*, 181–189.
- [LHR13] Lawhern, V., Hairston, W. D., & Robbins, K. (2013). DETECT: a MATLAB toolbox for event detection and identification in time series, with applications to artifact detection in EEG signals. *PloS One*, 8(4), e62944.
- [LLX+13] Liu, Y., Li, Z., Xiong, H., Gao, X., Wu, J., & Wu, S. (2013). Understanding and enhancement of internal clustering validation measures. *IEEE Transactions on Cybernetics*, 43(3), 982–994.
- [Lux07] Luxburg, U. Von. (2007). A Tutorial on Spectral Clustering. *Technical Report*, 17(March), 395–416.
- [Mac67] MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. *Proceedings of the Fifth Berkeley Symposium on Math. Statist. And Prob. 1*, 281–297.
- [MW47] Mann H. B. and Whitney, D. R. (1947). On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other. *Annals of Mathematical Statistics*, 18 (1): 50–60.
- [MB02] Maulik, U., & Bandyopadhyay, S. (2002). Performance evaluation of some clustering algorithms and validity indices. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(12), 1650–1654.
- [Mon09] Montgomery, D. (2009). *Introduction to statistical quality control*. John Wiley & Sons Inc.
- [MBV+07] Muthukrishnan, S., Berg, E. Van Den, & Wu, Y. W. Y. (2007). Sequential Change Detection on Data Streams. *Seventh IEEE International Conference on Data Mining Workshops (ICDMW 2007)*, 551–556.
- [NJW01] Ng, A. Y., Jordan, M. I., & Weiss, Y. (2001). On Spectral Clustering: Analysis and an algorithm.



- Advances in Neural Information Processing Systems*, 849–856.
- [Ohr01] Ohrnberger, M. (2001). *Continuous Automatic Classification of Seismic Signals of Volcanic Origin at Mt . Merapi , Java , Indonesia. Engineering.*
- [Pag54] Page, E. (1954). Continuous inspection schemes. *Biometrika*, 41(1), 100–115.
- [PSK06] Pang-Ning, T., Steinbach, M., & Kumar, V. (2006). *Introduction to data mining. Library of Congress.*
- [Pec08] Pecanka, J. (2008). Change Detection in Autoregressive Time. *Series Master Thesis, VU University Amsterdam.*
- [Pet79] Pettitt, A. (1979). A non-parametric approach to the change-point problem. *Applied statistics*, 28(2):126-135.
- [Ran71] Rand, W. M. (1971). Objective Criteria for the Evaluation of Clustering Methods. *Journal of the American Statistical Association*, 66(336), 846–850.
- [RV96] Real, R., & Vargas, J. M. (1996). Society of Systematic Biologists The Probabilistic Basis of Jaccard’s Index of Similarity. *Source: Systematic Biology*, 45(3), 380–385.
- [RR02] Rezek, I., & Roberts, S. J. (2002). Ensemble Hidden Markov models for biosignal analysis. In *International Conference on Digital Signal Processing, DSP* (Vol. 1, pp. 387–391).
- [RM05] Rokach, L., & Maimon, O. (2005). Clustering Methods. *Data Mining and Knowledge Discovery Handbook*, 321–352.
- [RTA11] Ross G. J., Tasoulis D. K., Adams N. M. (2011). Nonparametric monitoring of data streams for changes in location and scale. *Technometrics*, 53(4):379-389.

- [RA12] Ross, G. J. and Adams, N. M. (2012). Two nonparametric control charts for detecting arbitrary distribution changes. *Journal of Quality Technology*, 44(2).
- [Rou87] Rousseeuw, P. J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20(C), 53–65.
- [SK74] Scott, A. J. and Knott, M. (1974). A cluster analysis method for grouping means in the analysis of variance. *Biometrics*, 30(3):507-512.
- [SKK00] Steinbach, M., Karypis, G., & Kumar, V. (2000). A Comparison of Document Clustering Techniques. In *KDD workshop on text mining* (Vol. 400, pp. 1–2).
- [SKA+04] Stögbauer, H., Kraskov, A., Astakhov, S. A., & Grassberger, P. (2004). Least-dependent-component analysis based on mutual information. *Physical Review E*, 70, 1–17.
- [SG02] Strehl, A.; Ghosh, J. (2002). Cluster ensembles—A knowledge reuse framework for combining multiple partitions. *J. Mach. Learn. Res.*, 3, 583–617.
- [VHG+03] Vlachos M., Hadjieleftheriou M., Gunopulos D., Keogh E. (2003) Indexing multi-dimensional time-series with support for multiple distance measures. *Proceedings of the 9th International Conference on Knowledge Discovery & Data Mining (SIGKDD), Washington, DC*, 216–225.
- [Van79] Van Rijsbergen, C. J. (1979). *Information Retrieval, 2nd edition. Butterworths.*
- [Wal04] Wald, A. (2004). *Sequential Analysis, Dover Publications.*

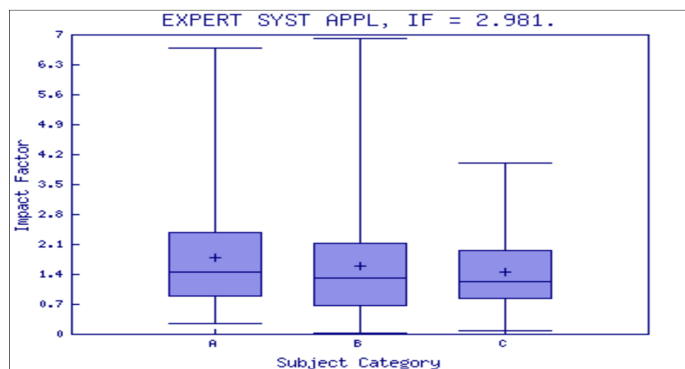
- [XB91] Xie, X. L., & Beni, G. (1991). A validity measure for fuzzy clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(8), 841–847.
- [YVC03] Ye, N., Vilbert, S., & Chen, Q. (2003). Computer intrusion detection through EWMA for autocorrelated and uncorrelated data. *IEEE Transactions on Reliability*, 52(1), 75–82.
- [ZK02] Zhao, Y., & Karypis, G. (2002). Evaluation of hierarchical clustering algorithms for document datasets. *Proceedings of the Eleventh International Conference on Information and Knowledge Management*, 515–524.
- [ZK04] Zhao, Y., & Karypis, G. (2004). Empirical and theoretical comparisons of selected criterion functions for document clustering. *Machine Learning*, 55(3), 311–331.

## Anexo 1

### A1.1 Expert Systems with Applications

Tres de las publicaciones se han realizado en la revista EXPERT SYSTEMS WITH APPLICATIONS.

Esta revista figura **en el Q1** en el 2015 JCR Science Edition con un factor de impacto de 2.981 (ver Figura A1.1). Es pues una revista reconocida en el campo de la informática aplicada. En concreto se encuentra situada en el número 27 de 257 dentro de la categoría ENGINEERING, ELECTRICAL & ELECTRONIC del JCR 2015 o en el número 6 de 82 de la categoría OPERATIONS & MANAGEMENT SCIENCES.



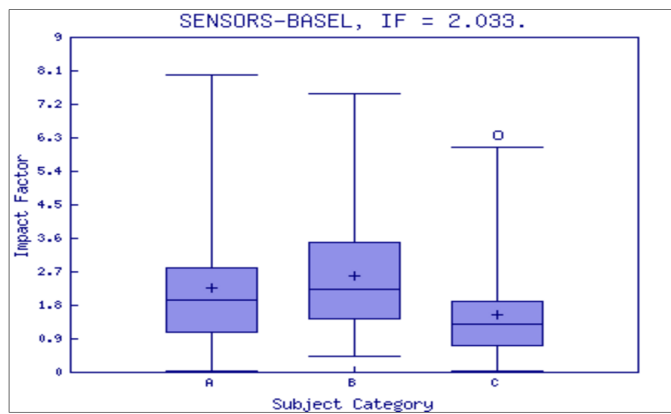
#### Key

- A - COMPUTER SCIENCE, ARTIFICIAL INTELLIGENCE
- B - ENGINEERING, ELECTRICAL & ELECTRONIC
- C - OPERATIONS RESEARCH & MANAGEMENT SCIENCE

Figura A1.1: Ranking de Expert Systems with Applications

## A1.2 Sensors

Una de las publicaciones se ha realizado en la revista SENSORS. SENSORS figura en el 2015 JCR Science Edition con un factor de impacto de 2.033. Es una revista situada **en el Q1**. En concreto se encuentra la número 12 de 56 dentro de la categoría de INSTRUMENTS and INSTRUMENTATION del JCR 2015 (ver Figura A1.2).



### Key

A - CHEMISTRY, ANALYTICAL

B - ELECTROCHEMISTRY

C - INSTRUMENTS & INSTRUMENTATION

Figura A1.2: Ranking de Sensors

## Anexo 2

Durante esta tesis se ha co-dirigido junto a la Dra. Raquel Dormido Canto y la Dra. Natividad Duro Carralero, los siguientes proyectos de fin de Máster del Departamento de Informática y Automática (UNED):

- **Proyecto 1:**

*Título:* Señales multicanal: análisis y filtrado de artefactos.

*Autor:* Luis Alberto Ramón Surutusa.

*Objetivo:* Las señales *EEG* poseen artefactos. La mayoría tienen un origen fisiológico. Por ejemplo, la actividad muscular de un guiño que se superpone a la señal *EEG*. El objetivo de este trabajo ha consistido en filtrar los artefactos fisiológicos mediante técnicas *ICA* a partir de la detección de los artefactos. La detección se ha realizado utilizando un método de clasificación supervisada.

- **Proyecto 2:**

*Título:* Estudio de métodos de determinación del número óptimo de componentes independientes en señales multicanal.

*Autor:* Daniel Mercader Rodríguez.

*Objetivo:* Existen varios métodos *ICA* que proporcionan las componentes independientes de una señal multicanal. Sin embargo el número de componentes independientes puede

no ser óptimo. El objetivo de este trabajo ha consistido en analizar los diferentes métodos presentes en la literatura para determinar el número óptimo de componentes independientes. Se ha realizado un “toolbox” en Matlab que permite estudiar y comparar los resultados de los métodos.

