CENTRO INTERNACIONAL DE ESTUDOS
DE DOUTORAMENTO E AVANZADOS
DA USC (CIEDUS)

TESIS DE DOCTORADO

ESTRATEGIAS ESTÁTICAS Y DINÁMICAS PARA EL AGRUPAMIENTO DE LATIDOS MEDIANTE ACUMULACIÓN DE EVIDENCIA

David González Márquez

ESCUELA DE DOCTORADO INTERNACIONAL

PROGRAMA DE DOCTORADO EN INVESTIGACIÓN EN TECNOLOXÍAS DA INFORMACIÓN

SANTIAGO DE COMPOSTELA

AÑO 2017

USC
UNIVERSIDADE
DE SANTIAGO
DE COMPOSTELA

USC
UNIVERSIDADE DE SANTIAGO DE COMPOSTELA

DECLARACIÓN DEL AUTOR DE LA TESIS

Estrategias estáticas y dinámicas para el agrupamiento de latidos mediante acumulación de evidencia

D. David González Márquez

Presento mi tesis, siguiendo el procedimiento adecuado al Reglamento, y declaro que:

1) *La tesis abarca los resultados de la elaboración de mi trabajo.*
2) *En su caso, en la tesis se hace referencia a las colaboraciones que tuvo este trabajo.*
3) *La tesis es la versión definitiva presentada para su defensa y coincide con la versión enviada en formato electrónico.*
4) *Confirmo que la tesis no incurre en ningún tipo de plagio de otros autores ni de trabajos presentados por mí para la obtención de otros títulos.*

En Santiago de Compostela, 23 de Octubre de 2017

Fdo David González Márquez

USC
UNIVERSIDADE
DE SANTIAGO
DE COMPOSTELA

USC
UNIVERSIDADE DE SANTIAGO DE COMPOSTELA

AUTORIZACIÓN DE LOS DIRECTORES DE LA TESIS

Estrategias estáticas y dinámicas para el agrupamiento de latidos mediante acumulación de evidencia

D. Paulo Félix Lamas
D. Abraham Otero Quintana

INFORMAN:

Que la presente tesis, corresponde con el trabajo realizado por D. ***David González Márquez****, bajo mi dirección, y autorizo su presentación, considerando que reúne los requisitos exigidos en el Reglamento de Estudios de Doctorado de la USC, y que como director de ésta no incurre en las causas de abstención establecidas en Ley 40/2015.*

En Santiago de Compostela, 23 de Octubre de 2017

Fdo Paulo Félix Lamas

Fdo Abraham Otero Quintana

USC
UNIVERSIDADE
DE SANTIAGO
DE COMPOSTELA

Life is like an ECG diagram.
If it goes smoothly, you are dead.

Internet

Mi corazón, mi corazón
es un músculo sano pero necesita acción.
Dame paz y dame guerra, y un dulce colocón
y yo te entregaré lo mejor.

Calamaro

USC
UNIVERSIDADE
DE SANTIAGO
DE COMPOSTELA

Agradecimientos

Me gustaría comenzar agradeciendo a mis directores Paulo Félix y Abraham Otero todo su trabajo, apoyo y dedicación. Han sido ellos los que me han acompañado, guiado y arrastrado (cuando hacía falta) y sin ellos nada de esto sería posible. Agradecer especialmente su paciencia y trato conmigo, sé que en este camino los tropiezos han sido míos y siempre habéis estado ahí para intentar apagar los fuegos. Gracias a Paulo por acogerme en su casa (el CiTIUS) y abrirme las puertas de par en par, poniendo todos los medios necesarios para que esta tesis haya podido seguir adelante. Gracias a Abraham por introducirme en el mundo de la investigación y llevarme de la mano todos estos años (y los que nos quedan). Todo agradecimiento queda corto.

Debo dar también las gracias a la profesora Ana Fred, por acogerme hasta en dos ocasiones en Lisboa y por sus consejos y dirección, sin ella esta tesis tampoco habría sido posible. Igualmente agradecer al Instituto Superior Técnico de Lisboa y al grupo "Pattern and Image Analysis – Lx" del Instituto de Telecomunicações que me recibieron durante mis dos estancias en Lisboa. También debo tener en cuenta a todos los compañeros y amigos que encontré en Lisboa, que hicieron de esos meses de estancia una experiencia para recordar (tanto que tuve que volver).

Dar las gracias al CiTIUS, que ha sido mi casa durante estos años y me ha permitido formarme y desarrollar mi tesis en un entorno óptimo. Entorno que no sería posible sin todas las personas que lo forman, desde los conserjes hasta el servicio técnico (especial mención a Jorge, todo un santo). También merecen una mención aparte los compañeros de laboratorio que me acogieron como uno más (a pesar de que el gallego no sea lo mío) y sin los cuales probablemente estaría interno en algún psiquiátrico. Agradecimiento extraordinario se merece Tino que por si no fuera poco con andar por este camino que compartimos tiene que ir arreglando los desastres que le dejo a mi paso.

Un agradecimiento merecen también las instituciones que han confiado en mí y han puesto los medios necesarios para que se realizara esta tesis, con especial mención a todos los contribuyentes que pagan sus impuestos. Gracias a la Universidad CEU San Pablo donde inicie mis estudios y mi carrera investigadora. A la Universidad de Santiago de Compostela en cuyo seno he realizado esta tesis. Agradecer al Ministerio de Educación de España su financiación por medio del programa de ayudas para la Formación del Profesorado Universitario (FPU AP2012-5053). A la Xunta de Galicia por su financiación por medio de una beca predoctoral (PRE/2012/389) y el proyecto MBEAT HOME EDITION. A la Universidad CEU San Pablo por su financiación a través de los proyectos PPC12/2014 y PCON10/2016. Al Ministerio de Economía, Industria y Competitividad por su financiación a través de los proyectos AI-SENIOR (TIN2009-14372-C03-03) y CARE-U (TIN2014-55183-R). Finalmente al banco Santander por financiar una de mis estancias doctorales en el IST de Lisboa por medio de una ayuda JPI para la movilidad de jóvenes investigadores.

A un nivel más personal me gustaría dar las gracias a mi familia y mis amigos por su apoyo durante esta etapa. Sé que muchas veces no ha sido fácil entender qué es lo que hacía (tampoco lo era para mí) o cuánto iba a tardar. Especialmente agradecer a mi madre todo su apoyo, ánimo y cariño; sé que siempre estás ahí y pocas veces lo agradezco. Finalmente, gracias al que siempre está ahí, y pocas veces lo vemos, gracias por todo.

"It's been one hell of a ride" pero literalmente, ha sido un largo ~~infierno~~ camino, una montaña rusa emocional, de emociones encontradas y una constante sensación de fatalidad inminente. Sin embargo, aquí estamos, final de etapa alcanzado. Momento de echar la vista atrás (ahora sí), reflexionar y sobretodo agradecer; a Dios, a la vida y a todas aquellas personas que me han acompañado y han sido parte de una u otra forma, esta tesis es nuestra tesis.

Tomar fuerzas y mirar hacia delante, el camino espera, esto... esto es solo el principio.

Noviembre de 2017

Summary of the thesis

In this thesis we present a set of solutions for the efficient processing and interpretation of the electrocardiogram. Since Pipberger digitized it for the first time in 1959, the ECG has attracted attention from multiple fields of science and engineering. Nowdays it has become a simple low-cost test for the study of cardiopathies which, since the Spanish flu epidemic of the 20th century, have been the first cause of death by disease in the world. Hence the interest in developing faster and better ECG analysis procedures.

The first stages of QRS analysis, detection and morphological heartbeat characterization, stand out for its importance. Based on them the cardiac cycle will be characterized and the underlying processes will be interpreted. Errors that occur in this first stage may invalidate the outcome of the interpretation. As consequence, in the clinical routine they still are performed by visual inspection of an expert cardiologist. In long duration recordings (including 24 hours) visual inspection is a tedious task and its outcome depends excessively on the cardiologist who performs it. Hence the interest in the development of computational analysis techniques.

The automatic detection of the QRS complex, the most significant feature of the heartbeat in the electrocardiogram, has been solved with an acceptable degree of satisfaction (around 99.7% correct detections on the reference databases). In the scientific literature, this fact is recognized and it appears as a priority goal the identification of the heartbeat according to its origin and propagation path in the cardiac muscle. In the bibliography there are two main alternatives to support the cardiologist in this task: classification techniques and clustering techniques.

In a classification technique an algorithm assigns a label to each heartbeat. Ideally, this label would be the same that the one assigned by the cardiologist. The bibliography includes a broad set of proposals that can perform heartbeat classification. In almost all the proposals an excellent behavior is observed on the databases on which the training was conducted.

However, they often present poor performance when presented with ECG waveforms from different patients. This is mainly due to the high dependency of these techniques on the training data set.

The clustering techniques divide the heartbeats in groups, according to some similarity measure. Afterwards, the cardiologist could label the groups, instead of labelling the heartbeats one by one. Bearing in mind that nowadays the morphological characterization is performed completely by the cardiologist, the clustering could be an important support, reducing considerably the time needed for ECG interpretation. Furthermore, we believe that any technique that leaves the final decision to the cardiologist is more susceptible of being used in the clinical routine than a fully automated solution. Therefore, this will be the focus of this thesis.

This thesis is organized as follows:

Chapter 1 This chapter is comprised of an introduction to electrocardiography concepts used in the thesis and a short review of the most relevant related literature. The electrical activity of the heart and its relationship with the trace of the ECG is explained briefly. Basic notions such as electrocardiographic leads, waves, intervals and segments are presented. The problem of arrhythmia recognition is stated, and finally, different techniques to automatically process the ECG are reviewed.

Chapter 2 Prior to the design of the clustering algorithm, the first issue that arises is the representation of the beat itself. In the literature there are three main approaches: using the raw signal, using a set of waveform characteristics obtained from the signal, or using a basis of functions. In this chapter, after discussing the advantages and disadvantages of these alternatives, a comprehensive study about the representation selected in this thesis is presented. The representation chosen is based on the Hermite basis functions due to its compactness and reliability. The calculation of the Hermite representation is explained, and a study on the optimal number of Hermite functions to represent the heartbeats according to AIC and BIC is presented.

Chapter 3 In this chapter an alternative to calculate the Hermite representation of heartbeats using GPUs is presented. The use of a GPU enables a considerable reduction in the time needed to calculate the representation, thereby freeing up the CPU to perform other tasks. First, an optimized version of the algorithm used to calculate the representation is presented. Afterwards, the algorithm is adapted and optimized to be executed by a

GPU using CUDA. The speedup is obtained by comparing both approaches in three different scenarios: processing short recordings, processing long recordings and online processing.

Chapter 4 The evidence accumulation clustering paradigm is presented in this chapter. The notion of an ensemble of partitions, how to create the different partitions, and how to combine them and extract the final partition are explained. The concept of negative evidence is also introduced. Based on this concept, a new clustering technique is developed: PN-EAC. This technique is applied to the problem of heartbeat clustering, and validated with the gold-standard database in the electrocardiographic domain, the MIT-BIH Arrhythmia Database. Three different strategies were tried: extracting positive evidence from all the available information simultaneously; extracting positive evidence from the information of each lead and the information derived from the distance between heartbeats independently, and extracting positive evidence from the information of each lead and negative evidence from the information derived from the distance between heartbeats. Finally, the technique PN-EAC was used to simultaneously extract evidence from up to 12 ECG leads at the same time; and the performance of the method is assessed with regard to different number of leads.

Chapter 5 In this chapter, a dynamic version of the PN-EAC technique, EPN-EAC, is presented. EPN-EAC can be used in scenarios which require online processing, providing partial results at anytime during the execution, as well as in the processing of very large recordings. The method is applied to the MIT-BIH Arrhythmia Database, using the same strategies as in the previous chapter. A comparison of results of the application of the strategies PN-AC and PN-EAC over several electrocardiographic databases, as well as other proposals of the literature is provided.

Conclusiones/Conclusions Finally, the main results of this thesis are summarized and conclusions are presented.

USC
UNIVERSIDADE
DE SANTIAGO
DE COMPOSTELA

Índice general

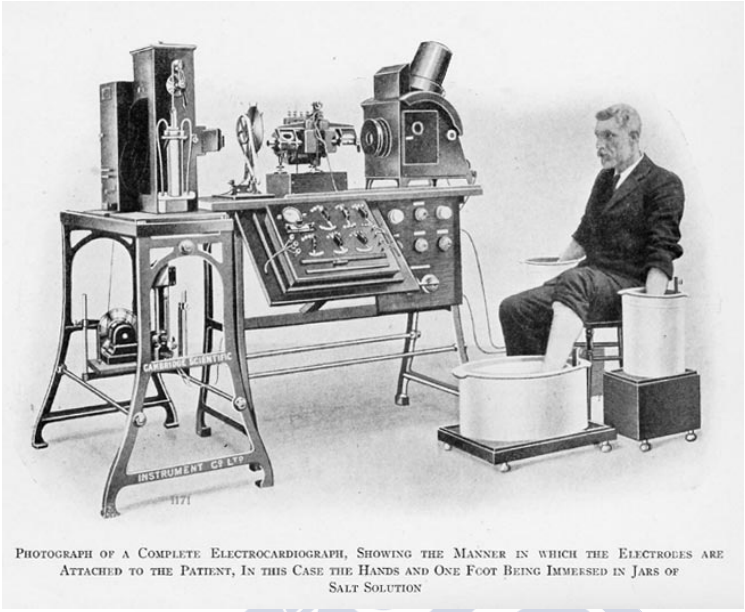
USC
UNIVERSIDADE
DE SANTIAGO
DE COMPOSTELA

Capítulo 1

Introducción

El campo de la electrocardiografía tiene su origen en los trabajos de Galvani [48] que observó las contracciones de unas ancas de rana al estimularlas con electricidad. Esta mal llamada en su momento "electricidad animal" marcó el inicio de un periodo de investigación sobre la interacción entre la electricidad y los sistemas biológicos presentes en el cuerpo humano. Serían posteriormente Matteucci, Kolliker y Muller, entre otros, a los que correspondería iniciar este nuevo campo y demostrar la relación entre el ciclo cardíaco y la actividad eléctrica del miocardio [98]. Este desarrollo teórico fue acompañado de avances en los instrumentos de medida de la mano de Waller [142] y Einthoven [118] (véase Figura 1.1).

Desde que por primera vez Pipberger digitalizó el electrocardiograma (ECG) en 1959 y diseñó los primeros programas para su análisis [114], el interés en esta señal fisiológica no ha parado de crecer, siendo en la actualidad una prueba fundamental en la rutina clínica. Su bajo coste y su sencillez lo convierten en una herramienta excepcional para el diagnóstico y seguimiento de enfermedades cardiacas, particularmente en países en vías de desarrollo. Además, su carácter no invasivo y la simplicidad de la instrumentación necesaria para su adquisición, hacen de él un candidato ideal para realizar una monitorización de larga duración durante la actividad rutinaria del paciente. Este interés en el ECG se ve reforzado por la alta mortalidad de las enfermedades cardiovasculares, que actualmente son la primera causa de muerte por enfermedad en el mundo [106]. Lejos de resolverse esta circunstancia, en el futuro cercano se prevé que continuará incrementando (véase Figura 1.2), especialmente en los países en vías de desarrollo, por los cambios en la dieta y estilo de vida derivados del mayor poder adquisitivo de sus ciudadanos [95].



PHOTOGRAPH OF A COMPLETE ELECTROCARDIOGRAPH, SHOWING THE MANNER IN WHICH THE ELECTRODES ARE ATTACHED TO THE PATIENT, IN THIS CASE THE HANDS AND ONE FOOT BEING IMMERSED IN JARS OF SALT SOLUTION

Figura 1.1: Antiguo electrocardiógrafo comercial construido en 1911 por la Cambridge Scientific Instrument Company. (Fuente: [24])

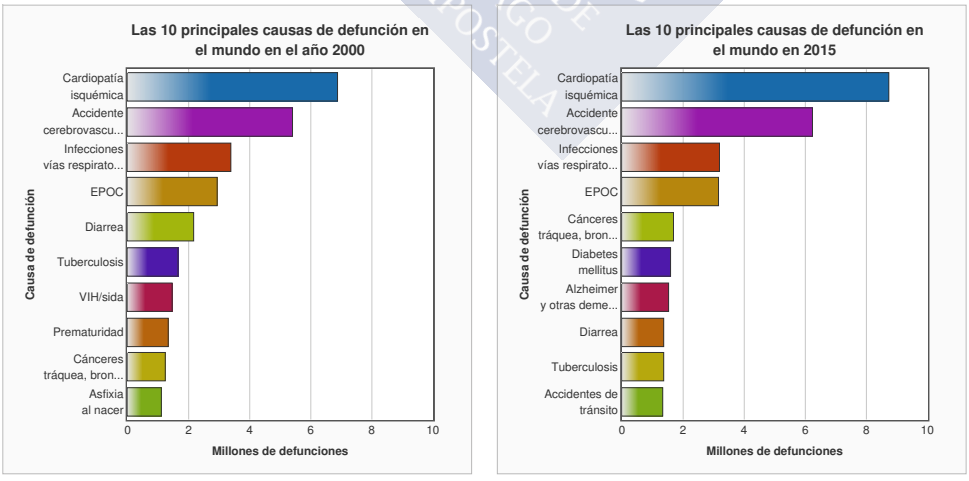


Figura 1.2: Principales causas de defunción en el mundo en 2000 y 2015. (Fuente: [106])

El electrocardiograma, además de servir como instrumento al servicio del diagnóstico de la patología cardiaca, es una fuente inagotable de información [92]. Gracias a su estudio es posible comprender mejor la compleja interacción entre los distintos procesos fisiopatológicos que concurren en las alteraciones del impulso eléctrico en el miocardio. Su análisis es objeto de innumerables trabajos científicos y la aparición de nuevas aplicaciones es continua: la estimación de la salud fetal en obstetricia [90], el seguimiento de pacientes crónicos como en el caso de la diabetes [25], la enfermedad pulmonar obstructiva crónica [64], la apnea-hipopnea del sueño [110], o incluso el diseño de nuevos fármacos [144], son solo algunos ejemplos.

En este contexto es indudable el interés en desarrollar nuevos procedimientos y métodos para el análisis computacional del ECG. Cualquier mejora en este sentido redundará en un gran beneficio por su aplicación en un amplio conjunto de escenarios que hacen uso de esta prueba médica. Estas mejoras podrían permitir una monitorización más precisa, ampliar el conocimiento sobre el miocardio y su patología y en última instancia podrían resultar en una mejora del tratamiento y la consecuente reducción en la pérdida de vidas humanas y mejora de calidad de vida.

Sin embargo, parece que en los últimos años los avances en el campo de la electrocardiografía se han ralentizado [4]. La instrumentación de adquisición, excepto pequeñas mejoras incrementales, no ha experimentado avances significativos a la hora de permitir obtener una mayor información acerca del funcionamiento electrofisiológico del miocardio. Por ello, es de esperar que los futuros avances no surgirán de un cambio en la tecnología de adquisición, sino de mejoras en el procesamiento del ECG, concretamente del procesado computacional.

Son de especial importancia en el análisis computacional del ECG las tareas que se sitúan en la primera fase de detección e identificación del latido cardíaco. A partir de estas tareas se caracterizará el ciclo cardiaco y se construirá el resto del análisis. Errores en esta primera fase pueden invalidar el resto de la interpretación realizada. Por ello estas tareas requieren una atención especial por parte del cardiólogo y en muchas ocasiones siguen siendo realizadas mediante inspección visual. Los registros Holter, habitualmente utilizados para el diagnóstico, pueden durar varios días; por tanto, su inspección visual resulta tediosa y el resultado de esta depende excesivamente del cardiólogo que la realiza. Además, debemos tener en cuenta la gran cantidad de pruebas de electrocardiograma que se realizan cada año (aproximadamente 200 millones en todo el mundo) [117].

En la siguiente sección realizaremos una introducción al funcionamiento electrofisiológico del corazón que servirá de base para comprender los problemas que serán abordados en la presente tesis doctoral.

1.1. Introducción a la electrocardiografía

El electrocardiograma es la representación gráfica de la actividad eléctrica del corazón en función del tiempo. Para su estudio es necesario tener una noción básica del funcionamiento del corazón, prestando especial atención al aspecto eléctrico. En esto consistirá la primera parte de esta introducción al campo de la electrocardiografía.

1.1.1. Funcionamiento eléctrico del corazón

El corazón es un músculo hueco que mantiene la circulación sanguínea por medio de contracciones y dilataciones rítmicas. Está compuesto por cuatro cavidades, dos ventrículos y dos aurículas, situadas en las mitades superior e inferior del corazón, respectivamente. En la Figura 1.3 se muestran dichas cavidades junto con el nódulo sinoauricular (SA), el nódulo auriculoventricular (AV) y el Haz de His (las fibras de Purkinje no se muestran en la figura por claridad, pero aparecerían a continuación de las ramas del Haz de His alrededor de los ventrículos). Como veremos a continuación, estas estructuras juegan un papel fundamental a la hora de comprender la propagación del estímulo eléctrico.

La aurícula derecha recibe la sangre venosa del cuerpo y la envía al ventrículo derecho, el cual la envía a su vez a los pulmones, lugar en el que la sangre se oxigena. Simultáneamente la aurícula izquierda recibe sangre oxigenada de los pulmones, que envía al ventrículo izquierdo para que la distribuya por todo el cuerpo. Para que este ciclo funcione de forma adecuada las contracciones se deben realizar de forma cíclica y ordenada. Para ello existe un sistema de estimulación y conducción eléctrica compuesto por fibras del músculo cardíaco especializadas en la generación y transmisión de impulsos eléctricos. El correcto funcionamiento de este sistema da lugar a un registro normal de la actividad eléctrica del corazón, como el que se observa en la Figura 1.4).

La acción de bombeo del corazón proviene de un sistema intrínseco de conducción eléctrica. El nódulo sinoauricular (también conocido como el “marcapasos del corazón”) está compuesto por células especializadas cuya velocidad de despolarización espontánea es superior al resto de las células cardíacas. El impulso eléctrico normalmente es generado en

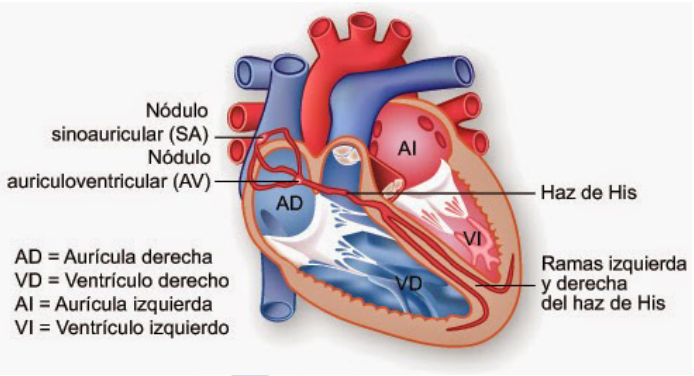


Figura 1.3: Representación del corazón con sus principales componentes. (Fuente: [19])

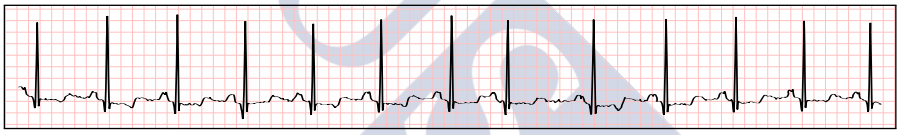


Figura 1.4: Fragmento de un ECG con latidos normales. (Fuente: MIT-BIH Arrhythmia Database, registro 100, entre 0:0:0 y 0:0:10)

este nódulo y propagado por el miocardio, causando la contracción ordenada de cada una de sus cavidades. Este nódulo genera un impulso eléctrico de 60 a 100 veces por minuto en condiciones normales. Dicho impulso se propagará por las aurículas provocando su contracción, y alcanzará el nódulo auriculoventricular. La velocidad de conducción eléctrica de las células de este nódulo es inferior a la asociada al tejido muscular en aurículas y ventrículos, lo que constituye una barrera en situaciones normales. Esto evita que los ventrículos desarrollen contracciones rápidas, desincronizadas e ineficientes como ocurre, por ejemplo, en la fibrilación ventricular (véase Figura 1.5). Cuando el impulso eléctrico alcanza este nódulo se produce un pequeño retraso (aproximadamente 0.1 segundos). Si el impulso se detiene en este punto se podría desembocar en una parada cardíaca, a menos que algunas células se despolaricen independientemente de dicho nódulo, originando una sucesión de latidos denominados “latidos de escape” (véase Figura 1.6).

En los ventrículos existen haces de células en torno a las cuales la velocidad de conducción del impulso eléctrico es muy superior a la asociada a las paredes musculares, lo que permite la contracción simultánea de toda la cavidad, fundamental para un bombeo adecuado. Estos

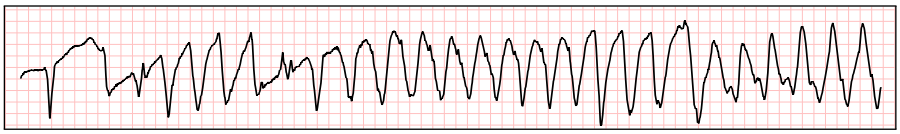


Figura 1.5: Fragmento de un ECG en el que se aprecia fibrilación ventricular. (Fuente: MIT-BIH Arrhythmia Database, registro 207, entre 0:0:40 y 0:0:50)

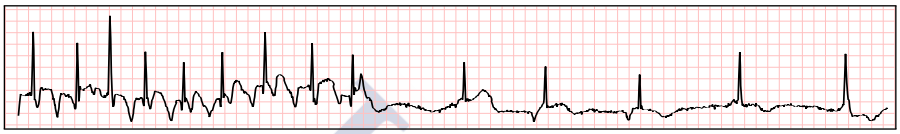


Figura 1.6: Fragmento de un ECG en el que aparecen una serie de latidos normales hasta que, aproximadamente en la mitad del fragmento, las distancias entre latidos se incrementan y aparecen latidos de escape. (Fuente: MIT-BIH Arrhythmia Database, registro 222, entre 0:12:36 y 0:12:46)

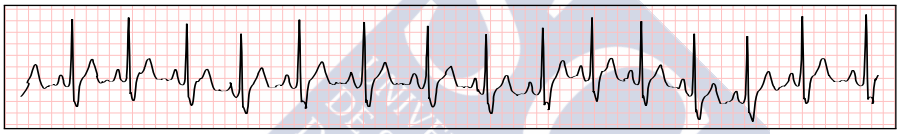


Figura 1.7: Fragmento de un ECG en el que aparecen latidos con bloqueo de rama derecha. (Fuente: MIT-BIH Arrhythmia Database, registro 212, entre 0:0:0 y 0:0:10)

haces integran el haz de His, que desciende desde el nódulo AV a lo largo de los ventrículos dividiéndose en dos ramas, derecha e izquierda. El bloqueo de estos haces origina situaciones de bloqueo de rama derecha o izquierda (véase Figura 1.7). A causa de la alta velocidad de conducción, en condiciones normales, el impulso se propaga rápidamente por el haz de His hasta alcanzar las fibras de Purkinje, generando la contracción ventricular. Este proceso se corresponde con la etapa de sístole que en situaciones de normalidad tiene una duración inferior a 100 ms. Tras la despolarización se produce la repolarización ventricular durante la cual las fibras musculares de los ventrículos se preparan para el próximo latido. Dado que los procesos de despolarización y repolarización se producen de forma coordinada en miles de células, las corrientes resultantes son suficientemente grandes como para generar diferencias de potencial de varios mV entre los distintos puntos de la superficie del cuerpo humano, las cuales pueden medirse con electrodos superficiales para obtener la señal de ECG.

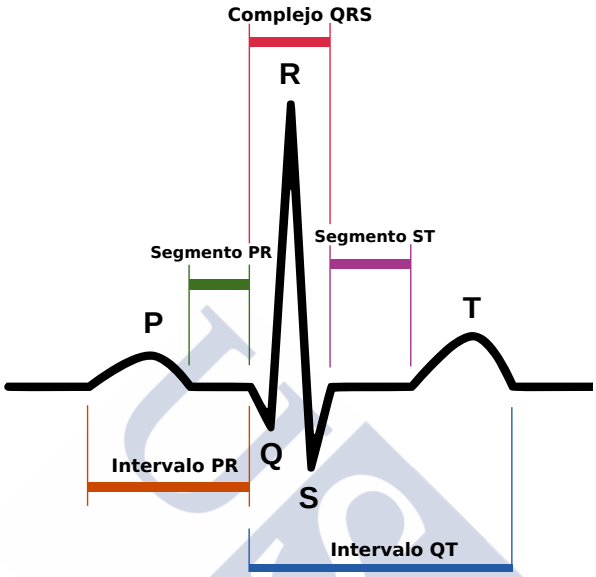


Figura 1.8: Imagen de un latido con las principales ondas y segmentos señalados. (Adaptado de [128])

1.1.2. Trazo eléctrico del latido cardíaco

Los elementos principales del trazado de un latido en el ECG son la onda P, el complejo QRS y la onda T (véase Figura 1.8). Estas ondas y los segmentos que las conectan tienen una relación directa con los eventos eléctricos que se producen en el corazón. La onda P corresponde a la despolarización auricular, como se aprecia en la Figura 1.9. La repolarización de las aurículas quedará eclipsada por la despolarización ventricular (complejo QRS) por lo que en un ECG normal no será posible visualizarla. El complejo QRS corresponde a la despolarización ventricular, que provoca la contracción de los ventrículos derecho e izquierdo (véase Figura 1.10). Finalmente, la onda T representa la repolarización de los ventrículos que recuperan su estado de reposo hasta la siguiente contracción (véase Figura 1.11). El intervalo QT mide la distancia entre el inicio del complejo QRS y el final de la onda T y se corresponde al tiempo entre la despolarización y la repolarización ventricular.

Un complejo QRS normal tiene una duración de entre aproximadamente 60 y 120 ms [71] y generalmente es la parte central y más visible del ECG. La masa cardíaca de los ventrículos es mucho mayor que la de las aurículas, ya que deben bombear la sangre hacia los pulmones

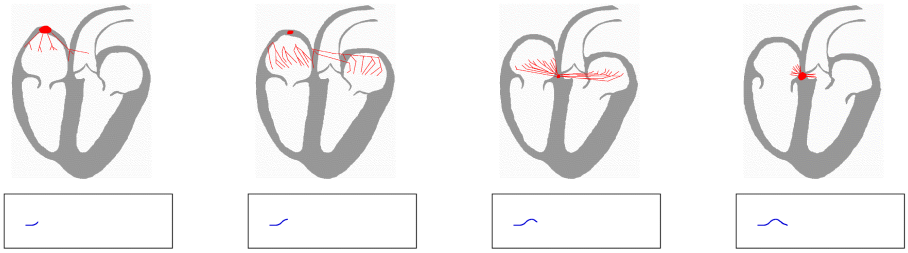


Figura 1.9: Representación de la despolarización de las aurículas y su correspondencia con la onda P del ECG. (Adaptado de [69])

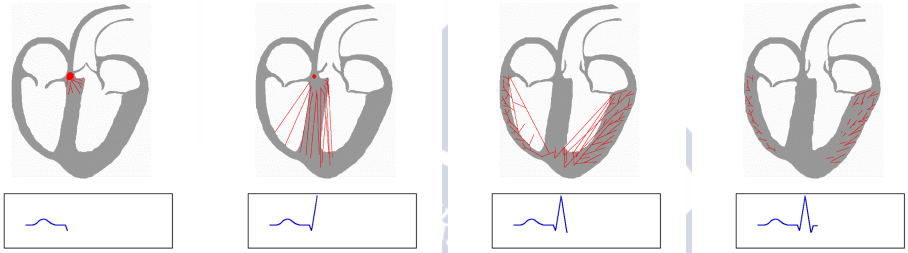


Figura 1.10: Representación de la despolarización de los ventrículos y de su correspondencia con el complejo QRS del ECG en un latido normal. (Adaptado de [69])

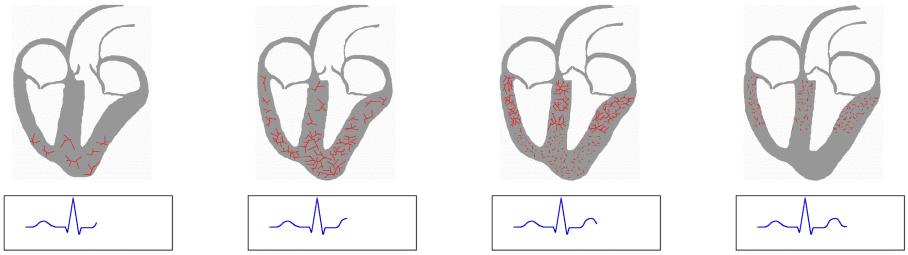


Figura 1.11: Representación de la repolarización de los ventrículos y su correspondencia con la onda T del ECG. (Adaptado de [69])

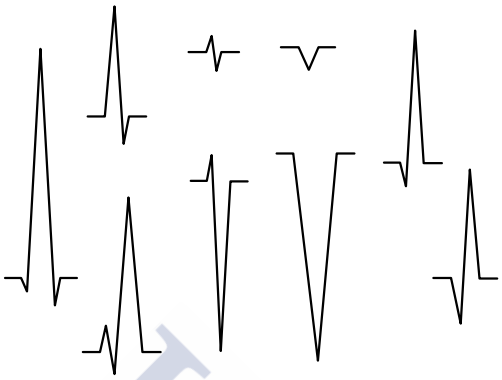


Figura 1.12: Ejemplos de distintos complejos QRS. (Adaptado de [97])

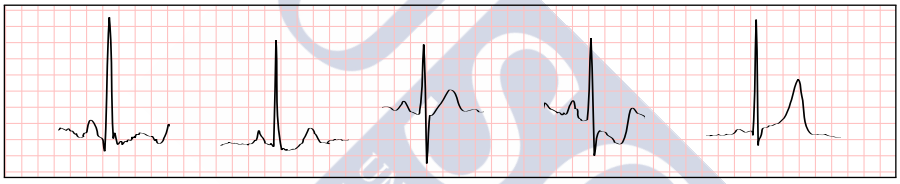


Figura 1.13: ECG mostrando distintas morfologías de latidos normales asociadas a diferentes pacientes en la misma derivación. (Fuente: MIT-BIH Arrhythmia Database)

o a todo el cuerpo, según se trate del ventrículo derecho o izquierdo, respectivamente. Por ello la despolarización es mucho más potente en el caso de los ventrículos, lo que convierte al complejo QRS en el elemento más distintivo del latido en el ECG. Además, debido a la alta conductividad del haz de His y las fibras de Purkinje, la velocidad de la despolarización en este caso es más alta, teniendo como consecuencia que las ondas del complejo QRS suelen ser angostas y en forma de pico, a diferencia de la forma redondeada de la onda P.

La duración, amplitud y morfología del complejo QRS proporcionan información valiosa sobre el estado del corazón y son útiles en el diagnóstico de arritmias cardíacas, anormalidades de la conducción y otros trastornos del corazón. El complejo QRS se corresponde con la combinación de tres ondas constituyentes del latido: Q, R y S; aunque no es necesario que aparezcan todas ellas para llamarlo así (véase Figura 1.12). Además, la forma del complejo depende en gran medida del paciente, no siendo igual para todos ellos (véase Figura 1.13), siendo posible incluso desarrollar técnicas de identificación biométricas basadas en ECG [86].

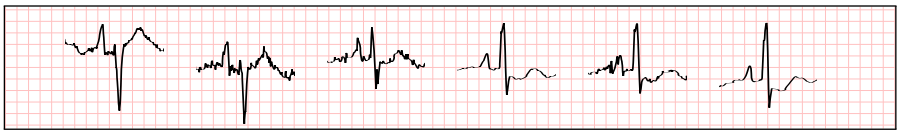


Figura 1.14: Fragmento de un ECG con latidos normales en los que se observa un cambio de morfología a lo largo del tiempo, entre cada latido hay una separación de 2 minutos. (Fuente: MIT-BIH Arrhythmia Database, registro 108, entre 0:00:00 y 0:10:00)

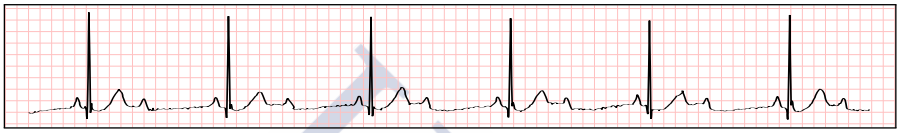


Figura 1.15: Fragmento de ECG en el que se observan varias ondas P adicionales, debido a un bloqueo de la conducción del impulso eléctrico. (Fuente: MIT-BIH Arrhythmia Database, registro 231, entre 0:01:45 y 0:01:55)

La morfología del complejo puede incluso cambiar con el tiempo en un mismo paciente, sin necesidad de que esté presente una patología cardíaca (véase Figura 1.14).

En la bibliografía médica se denomina arritmia a todo trastorno del ritmo cardíaco [132] que generalmente tendrá una repercusión visible en el ECG (forma del complejo QRS, duración, separación entre latidos, etc.). Una arritmia puede involucrar la presencia de actividad anómala en el músculo miocárdico, como por ejemplo contracciones sucesivas y desincronizadas de los ventrículos, reflejadas por un patrón sinusoidal en el ECG (sin distinguir complejos QRS ni ondas) como se ilustra en la Figura 1.5; o contracciones adicionales de las aurículas, reflejadas por dos ondas P seguidas (véase Figura 1.15). Las arritmias también pueden conllevar ausencia o demora en la actividad normal, como por ejemplo sucede cuando la señal eléctrica del latido no se propaga de un modo normal por la rama derecha del ventrículo (véase Figura 1.7). Las arritmias, clasificadas según el mecanismo que las produce, se dividen en dos grupos principales: trastornos en la generación del impulso eléctrico y trastornos en la conducción del impulso eléctrico.

En el primer grupo es posible distinguir entre trastornos del automatismo normal alterado, trastornos por automatismo anormal y trastornos por posdespolarizaciones tempranas o tardías [50]. En el miocardio, el automatismo es la capacidad de las células cardíacas de despolarizarse espontáneamente. Los trastornos del automatismo normal alterado son aquellos en los que están implicadas aquellas estructuras que normalmente

forman parte del automatismo del corazón pero que en este caso han alterado su funcionamiento habitual. Automatismo anormal se refiere a aquellos trastornos en los que está implicado un automatismo en una estructura ("foco ectópico") que normalmente no lo tiene. También se incluyen en este grupo los trastornos asociados con posdespolarizaciones. Estos trastornos, como su nombre indica, son despolarizaciones que ocurren desplazadas en el tiempo. En función de su relación temporal serán tempranas o tardías.

En los trastornos de la conducción del impulso eléctrico es posible distinguir dos tipos principales: bloqueo y reentrada. Los trastornos de bloqueo se producen cuando falla la propagación normal del impulso eléctrico (generalmente por razones anatómicas o funcionales), por ejemplo en el bloqueo auriculoventricular. La reentrada consiste en la reexcitación de zonas previamente despolarizadas. Durante la actividad eléctrica normal, el ciclo cardiaco se inicia en el nódulo sinoauricular y se propaga hasta activar todo el corazón, tras lo cual el impulso se extingue. Sin embargo, si un grupo aislado de fibras no se ha activado durante la onda inicial de despolarización, estas fibras pueden excitarse posteriormente. En este contexto, estas fibras podrían volver a excitar zonas previamente despolarizadas que ya se han recuperado de la despolarización inicial, generando lo que se conoce como excitación reentrante o reentrada.

1.1.3. Derivaciones del ECG

En el ECG una derivación es la medida de la actividad eléctrica del corazón mediante la diferencia de potencial entre dos puntos. Esta diferencia puede ser entre dos electrodos (derivación bipolar) o entre un punto virtual y un electrodo (derivaciones monopolares). Utilizando varias derivaciones para medir el impulso en distintas partes del cuerpo es posible obtener distintas perspectivas del mismo estímulo eléctrico.

La configuración de electrodos más utilizada en electrocardiografía es la llamada estándar de 12 derivaciones. Dicha configuración utiliza diez electrodos, cuatro conectados a las extremidades y 6 a lo largo del torso del paciente (véase Figura 1.16). Utilizando estos electrodos obtenemos las derivaciones precordiales (V1, V2, V3, V4, V5, V6), las derivaciones periféricas (I=LA-RA, II=LL-RA, III=LL-LA) y las derivaciones periféricas aumentadas (aVR=-(I+II)/2, aVL=I-II/2, aVF=II-I/2). Cada derivación muestra la actividad eléctrica del corazón desde un ángulo diferente; por tanto la información proveniente de las distintas derivaciones es complementaria (véase Figura 1.17).

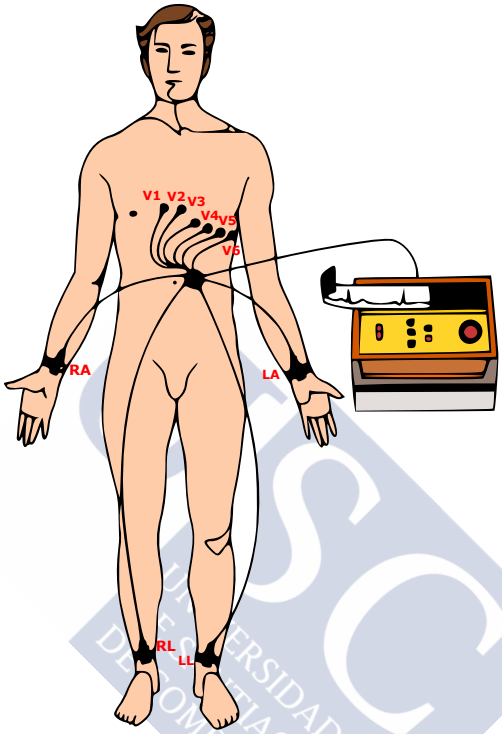


Figura 1.16: Esquema de la posición de los 10 electrodos en el cuerpo. (Adaptado de [88])

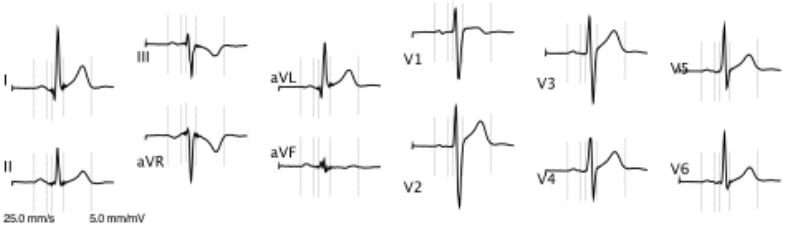


Figura 1.17: Ejemplo de la representación eléctrica de un mismo latido sobre las 12 derivaciones estándar. (Fuente: [127])

Figura 1.18: Fragmento de un ECG con dos derivaciones que ilustra cómo una patología, en este caso el aleteo auricular, puede reflejarse de forma distinta en cada derivación. (Fuente: MIT-BIH Arrhythmia Database, registro 202, entre 0:28:39 y 0:28:44)

Esto hace que ciertos matices se aprecien mejor en unas derivaciones que en otras ya que, para una derivación en concreto, parte de la actividad eléctrica puede no ser registrada o estar enmascarada [53] (véase Figura 1.18). De hecho, existen derivaciones específicamente diseñadas para detectar una patología concreta de forma sencilla [10]. Por ejemplo, la derivación Lewis (una variación de la derivación V5) [81] está especialmente indicada para detectar el aleteo auricular y las ondas P, incluso en taquicardia [32]. Es por esto que los cardiólogos usualmente se apoyan en la interpretación de varias derivaciones para el diagnóstico, comparándolas de forma paralela y combinando su información.

1.2. Identificación morfológica de latidos

En la bibliografía científica se reconoce que el problema de la detección del complejo QRS, el rasgo más significativo del latido cardiaco en el trazado electrocardiográfico, es un problema resuelto con un grado de satisfacción aceptable, y con una exactitud que en distintas aproximaciones ronda el 99.7 % [56][111][147] sobre las bases de datos de referencia. Sin embargo, aparece como un reto prioritario el de la identificación morfológica del latido. Esta identificación morfológica se reconoce como una ayuda importante para el experto cardiólogo

de cara a la interpretación del ECG. La interpretación requiere una gran cantidad de tiempo, especialmente en registros de larga duración, necesarios en aquellos casos en que los síntomas aparecen intermitentemente [73]. A mayor duración del registro, mayor el tiempo necesario para la interpretación y mayor la necesidad de soporte computacional.

Aunque la detección de arritmias es uno de los primeros pasos en el análisis del ECG, aportando información fundamental para el diagnóstico de una multitud de enfermedades cardíacas, en la práctica totalidad de las propuestas de la bibliografía se observa un comportamiento excelente sobre las bases de datos de referencia y un comportamiento mediocre cuando han de abordar el análisis de registros correspondientes a nuevos pacientes [63]. Estamos, por tanto, ante un problema abierto, y aún lejos de proporcionar una respuesta lo suficientemente satisfactoria. Entre las dificultades que se identifican en la resolución de este problema se encuentran: 1) la variabilidad de los procesos fisiológicos entre distintos pacientes, y aún en el mismo paciente; 2) el carácter estocástico de estos procesos; 3) la posibilidad de que múltiples procesos fisiológicos concurran simultáneamente, y sus múltiples interacciones; 4) la presencia de ruido y artefactos que enmascaran los procesos fisiopatológicos de interés; 5) la ausencia de modelos de funcionamiento del miocardio suficientemente completos y fiables, dejando en manos de la experiencia del cardiólogo la resolución del problema; y 6) el conocimiento tácito, subjetivo y difícilmente formalizable que constituye la experiencia del cardiólogo.

Dado que las arritmias se deben a cambios en el punto o el instante de la activación del impulso eléctrico y/o alteraciones en el camino de propagación, estas se reflejan visualmente en el ECG, afectando a la morfología del latido (véanse Figuras 1.5 y 1.7) o a la separación entre latidos (véase Figura 1.6). Es por ello que la identificación automática mediante técnicas computacionales de las distintas morfologías de latido presentes en un registro sería de gran ayuda al cardiólogo como paso previo a su interpretación. En la bibliografía aparecen distintas técnicas computacionales para esta labor, pudiendo distinguirse dos grandes tipos: técnicas de clasificación y técnicas de agrupamiento [21].

1.2.1. Clasificación de latidos

En la bibliografía la tarea de clasificación de latidos consiste generalmente en asignar a cada latido una etiqueta identificando el origen del latido. Idealmente dicha etiqueta sería la misma que asignaría un cardiólogo al latido. En [37] la Association for the Advancement of Medical Instrumentation (AAMI), entre otras recomendaciones para la evaluación del

rendimiento de algoritmos de análisis del ritmo cardíaco, propone una división de los latidos en 5 tipos: latido normal (N), supraventricular (S), ventricular (V), fusión (F) e indeterminado (Q). Esta clasificación ha sido aceptada y utilizada por la mayor parte de los algoritmos de clasificación, convirtiéndose en un estándar de facto.

La mayoría de los clasificadores requieren una etapa de entrenamiento para la cual es necesario contar con un conjunto de datos suficientemente representativo y etiquetado, lo que en el caso que nos atañe supondría contar con un conjunto amplio de latidos ya preinterpretados por el cardiólogo. En la bibliografía aparece una multitud de técnicas de aprendizaje automático aplicadas a esta tarea [87], como el Análisis Discriminante Lineal de Fisher [30], los Modelos Ocultos de Markov [8], Filtros Bayesianos [122] o las Máquinas de Vectores Soporte [107]; y técnicas que proceden de la inteligencia artificial, como las Redes de Neuronas Artificiales [40], los Algoritmos Evolutivos [54], o la Teoría de la Resonancia Adaptativa [23].

El principal punto débil de las técnicas de clasificación es la fuerte dependencia del resultado con el conjunto de entrenamiento y la diversidad presente en dicho conjunto. Las diferencias entre pacientes hacen que no se pueda asumir que un clasificador entrenado en un conjunto de datos producirá un resultado válido en nuevos pacientes [85] (véase Figura 1.13). Por ello, en ocasiones se opta por entrenar un clasificador sobre una amplia base de datos de latidos obtenidos de múltiples pacientes, incorporando a posteriori otra etapa de entrenamiento específica para cada paciente sobre un conjunto de latidos anotados. Las propuestas de la bibliografía con mejores resultados siguen este método, acercándose más a la idea de un clasificador asistido que a un clasificador completamente automático [31][63][72]. Incluso en este escenario, las diferencias observadas a lo largo del tiempo en la morfología de los latidos de un mismo paciente hacen que no se pueda asumir que la morfología de un tipo de latido no sufrirá alteraciones (véase Figura 1.14), pudiendo incluso aparecer nuevos tipos de latidos, no presentes en el conjunto de entrenamiento [67]. Otra dificultad para los clasificadores es que latidos con diferencias significativas a nivel morfológico pueden corresponderse con un mismo tipo de latido para el cardiólogo (véase Figura 1.13) [143]. Esto complica el diseño de un clasificador que use las mismas clases que el cardiólogo. Por otro lado, para mantener el número de dimensiones del espacio de características usado en la representación del latido en unos márgenes razonables, la gran mayoría de los trabajos de la literatura emplean una o dos derivaciones como fuente de dichas características [30][68][77]. Si en la fase de entrenamiento del clasificador se usaron

latidos provenientes siempre de las mismas derivaciones, la generalidad del clasificador al aplicarse sobre derivaciones diferentes no está garantizada, por lo que su uso se ve restringido a aquellos escenarios donde se dispone de las mismas derivaciones sobre las cuales se realizó el entrenamiento. Por contra, usar en el entrenamiento latidos provenientes de derivaciones diferentes resulta un reto significativo por la gran variabilidad morfológica de un mismo latido registrado en diferentes derivaciones (véase Figura 1.17).

1.2.2. Agrupamiento de latidos

La tarea del agrupamiento consiste en particionar un conjunto de datos en grupos, de tal modo que dicha partición refleje la estructura subyacente a los datos. En este caso no es necesario un conjunto de entrenamiento con latidos etiquetados, ni ningún tipo de entrenamiento específico para cada paciente. Al no existir una etapa previa de entrenamiento, tampoco existe dependencia con ninguna derivación concreta a la hora de realizar el análisis del ECG. Un agrupamiento exitoso permite resumir de forma eficaz registros de ECG, presentando al cardiólogo una serie de grupos donde los latidos pertenecientes a un mismo grupo presentan características similares; esto es, un mismo origen y camino de propagación del impulso eléctrico por el miocardio. El agrupamiento permite analizar las distintas familias morfológicas, algunas de ellas claramente asociadas a comportamientos patológicos, establecer las familias de normalidad, e identificar cuándo se producen cambios morfológicos, para luego proceder, mediante inspección del registro, a analizar su contexto temporal. En este escenario el hecho de que latidos con morfologías diferentes se correspondan con un mismo tipo de latido no supone un reto, ya que el algoritmo puede encontrar grupos diferentes para cada morfología del mismo tipo de latido, y el cardiólogo puede establecer la correspondencia de varios grupos con un mismo tipo.

El uso de técnicas de agrupamiento para la identificación morfológica de latidos resuelve buena parte de las desventajas del uso de clasificadores, a costa de requerir la intervención del cardiólogo para realizar la interpretación final de los grupos. Mientras que con un algoritmo de clasificación es teóricamente posible proporcionar directamente una interpretación de cada latido cardiaco (implícita en la etiqueta asignada), en el agrupamiento será la interacción con el cardiólogo la que permita llegar a dicha interpretación [93]. Si tenemos en cuenta que en la actualidad en la rutina clínica siempre se lleva a cabo un proceso de inspección visual del ECG, el escenario en el cual en vez de analizar directamente el ECG simplemente se valoran algunos representantes de un número reducido de grupos de latidos constituye una mejora

notable. Además, desde nuestro punto de vista, una aproximación donde la decisión final está en manos del cardiólogo, como es el caso del agrupamiento, es más susceptible de ser incorporada a la rutina clínica que una solución que trate de automatizar todo el proceso. Es por ello que en esta tesis doctoral se hace una apuesta por las técnicas de agrupamiento como una herramienta de apoyo al cardiólogo en la identificación morfológica de latidos.

En el agrupamiento de latidos es habitual reducir la representación del latido a su rasgo más significativo: el complejo QRS. El motivo principal es la difícil identificación y caracterización de todos los constituyentes del latido: en particular, la forma y tamaño de las ondas P y T hacen que en ocasiones aparezcan enmascaradas por el ruido de la señal de ECG. Incluir estas ondas podría hacer que en el agrupamiento se encontraran diferencias derivadas de inexactitudes en la representación y no de distintas morfologías. Limitando la representación del latido al complejo QRS se gana en sencillez y en uniformidad en la representación, además de reducir el tamaño del espacio de características.

En la bibliografía podemos encontrar múltiples trabajos que realizan agrupamiento de latidos, de los que aquí revisamos los más relevantes. El trabajo más referenciado en la bibliografía es [77]. En este trabajo se realiza un agrupamiento basado en la morfología del complejo QRS y en la distancia entre latidos. Se utilizó la base de datos MIT-BIH Arrhythmia Database al completo, que fue remuestreada a 1000 Hz. La detección de latidos se realizó mediante un algoritmo propio que detecta la posición de la onda R de cada latido. Posteriormente, se extrajo una ventana fija de 200 ms de señal alrededor de la posición dada por el anotador para representar cada complejo QRS. El agrupamiento posterior se realiza por medio de mapas auto-organizados (SOM). Los mapas auto-organizados están compuestos por varias neuronas que evolucionan mediante aprendizaje por competición, utilizando una función de vecindad para preservar las propiedades topológicas del espacio de representación. En [77] se utilizaron 25 neuronas, dando lugar a 25 grupos por registro. Los latidos fueron divididos en 16 tipos diferentes, excluyendo los latidos que solo contienen onda P, obteniendo en el mejor de los casos un error total cercano al 1.5 %.

Otra técnica orientada a reducir el número de latidos que tiene que revisar el cardiólogo se presenta en [27]. En este caso la técnica consta de dos etapas: en la primera se intenta reducir el número de latidos a agrupar calculando la similitud entre latidos, en la segunda se realiza el agrupamiento propiamente dicho, con dos alternativas: el algoritmo K-means y el algoritmo Max-Min. La técnica fue probada sobre 27 registros de ECG de la MIT-BIH Arrhythmia Database, que fueron elegidos intentando abarcar el mayor número de patologías

posible. Para extraer la información del ECG se utiliza una ventana de señal variable, alrededor de la anotación proporcionada por un detector de latidos. Posteriormente los latidos son normalizados y anotados manualmente por cardiólogos, lo que permite la verificación de los resultados. En la primera etapa el algoritmo redujo el número de latidos de 27412 a 1026. Estos latidos fueron posteriormente agrupados en la segunda etapa, logrando en el mejor de los casos un 7 % de error con el algoritmo K-means.

Otras aproximaciones más recientes también utilizan el algoritmo Max-Min para realizar el agrupamiento [134]. Este trabajo presenta una técnica novedosa para analizar la información del ECG a partir de un análisis simbólico. Para ello en una primera fase los latidos son detectados y agrupados. Posteriormente a cada grupo distinto de latidos se le asignará un símbolo identificativo. Con ello se consigue aumentar la abstracción del análisis, pasando de analizar la señal de ECG a analizar una secuencia de símbolos. Esto reduce la cantidad de datos y la dimensionalidad, facilitando el descubrimiento de patrones. Mediante el análisis simbólico es posible detectar cambios de ritmo y patrones temporales, entre otras anormalidades, y ver su relación con el estado del paciente y los fenómenos fisiopatológicos que concurren en el miocardio. El algoritmo de agrupamiento obtiene un error sobre la base de datos MIT-BIH Arrhythmia Database del 1.37 %, utilizando 6 tipos de latidos y un número variable de grupos por registros (mediana de 22 grupos por registro).

En [26] se propone una técnica de agrupamiento específica para la detección y el análisis de latidos ventriculares prematuros. La técnica utiliza una única derivación del ECG, que es filtrada para eliminar el ruido de alta frecuencia y la deriva de la línea base. Se aplica un detector de complejos QRS para localizar la onda R. Alrededor de esta posición se extrae una ventana variable de señal, dependiente de la distancia entre latidos. Para permitir comparar latidos de distinta longitud se utiliza DTW (Dynamic Time Warping) y el agrupamiento se basa en una combinación de algoritmos jerárquicos y algoritmos basados en centroides. En la técnica el número de grupos es ajustado dinámicamente para cada registro. La técnica fue aplicada a los registros de la base de datos MIT-BIH Arrhythmia Database, pero únicamente a los latidos etiquetados como normales o ventriculares prematuros. Para cada registro se utilizaron entre 2 y 30 grupos, con un porcentaje de error por registro menor al 1 % en la mayor parte de los registros.

En [75] se propone una técnica que combina agrupamiento y clasificación de complejos QRS, utilizando para el agrupamiento algoritmos basados en optimización mediante colonia de hormigas y para la clasificación redes neuronales y el método de los k vecinos más

cercanos. Se utiliza un algoritmo adaptativo para detectar los complejos QRS, que son posteriormente normalizados. Después se realiza el agrupamiento y basándose en los resultados del agrupamiento se lleva a cabo la clasificación. También se produce una retroalimentación de la clasificación al agrupamiento por lo que en la etapa de clasificación se puede decidir volver a recalcular el agrupamiento. La técnica fue aplicada a un subconjunto de la base de datos MIT-BIH Arrhythmia Database obteniendo, en el mejor de los casos, una sensibilidad media del 94.4 %.

Finalmente, en [20] se propone un algoritmo dinámico de agrupamiento de latidos basado en plantillas que representan los grupos que se crean, se unen, se eliminan y evolucionan con el tiempo. En este trabajo se utilizó la base de datos MIT-BIH Arrhythmia Database completa. Para extraer los complejos QRS se utiliza una ventana de tamaño fijo de 200 ms. Al realizar comparaciones entre latidos se utiliza DTW para alinearlos. En el agrupamiento cada grupo se representa por una plantilla. El algoritmo se ejecuta dinámicamente, por lo que para cada nuevo latido se debe decidir si unirlo a un grupo ya existente o crear un grupo nuevo. Posteriormente, se produce una actualización de los grupos afectados y puede producirse alguna fusión de grupos. Las comparaciones entre los latidos y las plantillas se realizan basándose en el análisis y comparación de puntos relevantes en el trazado del ECG. Adicionalmente, además de comparar la morfología se puede añadir una etapa al agrupador en la que se compara también características derivadas de la distancia entre latidos. El algoritmo propuesto tiene una latencia mínima, siendo el primero de la bibliografía en proponer una solución completa que puede funcionar en tiempo real. Mediante la aplicación de este algoritmo a la base de datos MIT-BIH Arrhythmia Database completa se obtiene un error del 1.44 % utilizando los 17 tipos de la base de datos.

1.3. Bases de datos de electrocardiografía

En esta sección introduciremos las bases de datos electrocardiográficas que se usarán para validar las técnicas desarrolladas en esta tesis.

1.3.1. MIT-BIH Arrhythmia Database

Se utilizará la base de datos más referenciada en la literatura de identificación de arritmias: la MIT-BIH Arrhythmia Database [96]. La gran variedad de pacientes, los distintos tipos de

latidos y la gran cantidad de anotaciones, han convertido esta base de datos en un "gold-standard" [16][30][31][77][108][112][152].

Esta base de datos está compuesta de 48 registros de electrocardiograma obtenidos de 47 pacientes distintos. Cada registro consta de dos derivaciones entre las siguientes: MLII, V1, V2, V3, V4 y V5. Los registros están digitalizados a una frecuencia de muestreo de 360 Hz con una resolución de 11 bits. La base de datos no se debe considerar como una muestra representativa de la población ya que los registros fueron seleccionados cuidadosamente para intentar abarcar la mayor variedad de trastornos cardiacos posible. Cada latido fue anotado por al menos dos cardiólogos, siendo aproximadamente el 68 % de ellos considerados normales mientras que el resto se dividieron entre 16 tipos de latidos anormales (ver Tabla 1.1). Los latidos en los que solo aparece la onda P (p) no serán utilizados en este trabajo al no tener complejo QRS.

La base de datos MIT-BIH Arrhythmia Database es anterior a la recomendación de la AAMI para etiquetar latidos [37]. Por ello utiliza sus propias etiquetas, dividiendo los latidos en 17 tipos distintos. La correspondencia entre las etiquetas de la base de datos MIT-BIH Arrhythmia Database y las etiquetas de la AAMI se muestra en la Tabla 1.2 (para evitar confusiones se mantienen los nombres originales en inglés de los tipos de latidos). En [30] se propone una correspondencia ligeramente distinta, que ha sido adoptada y utilizada por varios autores [33][76][85]. Sin embargo, siguiendo las recomendaciones de la AAMI, la correspondencia propuesta aquí es la correcta. Latidos auriculares de escape (e) y latidos nodales de escape (j) deberían ser asignados al tipo S de la AAMI en vez de al tipo N como se hace en [30].

1.3.2. St.-Petersburg Institute of Cardiological Technics 12-lead Arrhythmia Database

La base de datos "St.-Petersburg Institute of Cardiological Technics 12-lead Arrhythmia Database" (INCARTDB) consta de 75 registros muestreados a una frecuencia de 257 Hz provenientes de 32 pacientes distintos. La variedad de trastornos cardiacos no es tan amplia como en la MIT-BIH Arrhythmia Database, pero a cambio tiene un número mayor de registros y más derivaciones. Es una de las pocas bases de datos de la bibliografía con 12 derivaciones y anotada latido a latido, por lo que su uso está muy extendido [7][55][84][89].

Los registros fueron obtenidos de pruebas Holter realizadas para detectar la arterioesclerosis coronaria, por lo que en muchos de ellos se observan latidos ventriculares

Tabla 1.1: Número de latidos clasificados por tipo en la base de datos MIT-BIH para cada registro. (Fuente: [104])

Record	.	L	R	A	a	J	S	V	F	!	e	j	E	P	f	p	Q
100	2239	-	-	33	-	-	-	1	-	-	-	-	-	-	-	-	-
101	1860	-	-	3	-	-	-	-	-	-	-	-	-	-	-	-	2
102	99	-	-	-	-	-	-	4	-	-	-	-	-	2028	56	-	-
103	2082	-	-	2	-	-	-	-	-	-	-	-	-	-	-	-	-
104	163	-	-	-	-	-	-	2	-	-	-	-	-	1380	666	-	18
105	2526	-	-	-	-	-	-	41	-	-	-	-	-	-	-	-	5
106	1507	-	-	-	-	-	-	520	-	-	-	-	-	-	-	-	-
107	-	-	-	-	-	-	-	59	-	-	-	-	-	2078	-	-	-
108	1739	-	-	4	-	-	-	17	2	-	-	1	-	-	-	11	-
109	-	2492	-	-	-	-	-	38	2	-	-	-	-	-	-	-	-
111	-	2123	-	-	-	-	-	1	-	-	-	-	-	-	-	-	-
112	2537	-	-	2	-	-	-	-	-	-	-	-	-	-	-	-	-
113	1789	-	-	-	6	-	-	-	-	-	-	-	-	-	-	-	-
114	1820	-	-	10	-	2	-	43	4	-	-	-	-	-	-	-	-
115	1953	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
116	2302	-	-	1	-	-	-	109	-	-	-	-	-	-	-	-	-
117	1534	-	-	1	-	-	-	-	-	-	-	-	-	-	-	-	-
118	-	-	2166	96	-	-	-	16	-	-	-	-	-	-	-	10	-
119	1543	-	-	-	-	-	-	444	-	-	-	-	-	-	-	-	-
121	1861	-	-	1	-	-	-	1	-	-	-	-	-	-	-	-	-
122	2476	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
123	1515	-	-	-	-	-	-	3	-	-	-	-	-	-	-	-	-
124	-	-	1531	2	-	29	-	47	5	-	-	5	-	-	-	-	-
200	1743	-	-	30	-	-	-	826	2	-	-	-	-	-	-	-	-
201	1625	-	-	30	97	1	-	198	2	-	-	10	-	-	-	37	-
202	2061	-	-	36	19	-	-	19	1	-	-	-	-	-	-	-	-
203	2529	-	-	-	2	-	-	444	1	-	-	-	-	-	-	-	4
205	2571	-	-	3	-	-	-	71	11	-	-	-	-	-	-	-	-
207	-	1457	86	107	-	-	-	105	-	472	-	-	105	-	-	-	-
208	1586	-	-	-	-	-	2	992	373	-	-	-	-	-	-	-	2
209	2621	-	-	383	-	-	-	1	-	-	-	-	-	-	-	-	-
210	2423	-	-	-	22	-	-	194	10	-	-	-	1	-	-	-	-
212	923	-	1825	-	-	-	-	-	-	-	-	-	-	-	-	-	-
213	2641	-	-	25	3	-	-	220	362	-	-	-	-	-	-	-	-
214	-	2003	-	-	-	-	-	256	1	-	-	-	-	-	-	-	2
215	3195	-	-	3	-	-	-	164	1	-	-	-	-	-	-	-	-
217	244	-	-	-	-	-	-	162	-	-	-	-	-	1542	260	-	-
219	2082	-	-	7	-	-	-	64	1	-	-	-	-	-	-	133	-
220	1954	-	-	94	-	-	-	-	-	-	-	-	-	-	-	-	-
221	2031	-	-	-	-	-	-	396	-	-	-	-	-	-	-	-	-
222	2062	-	-	208	-	1	-	-	-	-	-	212	-	-	-	-	-
223	2029	-	-	72	1	-	-	473	14	-	16	-	-	-	-	-	-
228	1688	-	-	3	-	-	-	362	-	-	-	-	-	-	-	-	-
230	2255	-	-	-	-	-	-	1	-	-	-	-	-	-	-	-	-
231	314	-	1254	1	-	-	-	2	-	-	-	-	-	-	-	2	-
232	-	-	397	1382	-	-	-	-	-	-	-	1	-	-	-	-	-
233	2230	-	-	7	-	-	-	831	11	-	-	-	-	-	-	-	-
234	2700	-	-	-	-	50	-	3	-	-	-	-	-	-	-	-	-

Tabla 1.2: Correspondencia entre las anotaciones de la MIT-BIH Arrhythmia Database y las recomendadas por la AAMI.

Anotación AAMI	**Tipo de latido MIT-BIH (código)**
N	Normal beat (N)
	Left bundle branch block beat (L)
	Right bundle branch block beat (R)
S	Aberrated atrial premature beat (a)
	Supraventricular premature beat (S)
	Atrial premature beat (A)
	Nodal (junctional) premature beat (J)
	Nodal (junctional) escape beat (j)
	Atrial escape beat (e)
V	Ventricular flutter wave (!)
	Ventricular escape beat (E)
	Premature ventricular contraction (V)
F	Fusion of ventricular and normal beat (F)
Q	Paced beat (/)
	Unclassifiable beat(Q)

ectópicos. Ninguno de los pacientes tiene marcapasos y la edad media de los pacientes es de 58 años. Cada registro es un extracto de 30 minutos de duración y contiene las 12 derivaciones estándar. Todos los registros fueron anotados por un algoritmo automático y posteriormente estas anotaciones fueron corregidas manualmente. Las anotaciones de esta base de datos siguen la clasificación de latidos propuesta por la AAMI.

Capítulo 2

Representación de complejos QRS utilizando funciones de Hermite

La representación computacional de los datos que forman parte de un problema tiene un impacto considerable a la hora de resolver dicho problema. La elección de una representación inadecuada puede complicar e incluso impedir su resolución. En el caso que nos atañe, el agrupamiento morfológico de latidos, debemos elegir una representación para los latidos que, siendo lo más sencilla y compacta posible, permita resolver satisfactoriamente el problema del agrupamiento. Este primer paso, previo al análisis, influirá en el resto del proceso. Errores en esta fase pueden invalidar las subsecuentes fases y por tanto el resultado del análisis. Si la representación elegida no captura suficientes características del latido para permitir su correcto agrupamiento no habrá forma de corregir la elección en etapas posteriores. Si la representación captura matices irrelevantes o innecesarios su almacenamiento y procesado se complicará, y estos matices pueden añadir ruido que dificulte el agrupamiento correcto.

En la bibliografía podemos encontrar diversas propuestas de representación del latido, pero la mayoría se pueden agrupar en tres aproximaciones:

1. **Señal**. Para representar el latido se utiliza un fragmento del electrocardiograma en forma de señal digital, es decir, una señal discreta en tiempo discreto. Esta señal puede haber sufrido algún tipo de procesamiento previo con el objetivo de eliminar el ruido de la red eléctrica, suprimir la deriva de línea base o eliminar artefactos de alta frecuencia [62][131]. En algunos casos también se aplica una etapa de submuestreo

para destacar los rasgos más significativos, o para disminuir la alta dimensionalidad de la representación [30]. Esta es la representación más fiel del ECG e, idealmente, contiene además de la información de la proyección de la actividad eléctrica del miocardio, información sobre la actividad eléctrica de otros fenómenos fisiológicos que se superponen a los específicos del miocardio. Sus puntos débiles son la vulnerabilidad ante la presencia de ruido y artefactos, que no siempre resultan fáciles de eliminar, y la elevada dimensionalidad del vector de representación resultante [12]. Por ejemplo, para representar el complejo QRS lo habitual sería considerar al menos 200 ms de señal que, con una frecuencia de muestreo de 360 Hz, implica 72 muestras para representar un complejo QRS. Este valor habría que multiplicarlo por el número de derivaciones que se vayan a utilizar, que habitualmente no es menor de dos.

2. **Características**. Se representa el latido mediante un vector de características derivadas de la señal muestreada que representa el latido. Estas características pueden ser de diversa naturaleza, como la media o la desviación típica, la curtosis o medidas derivadas de la caracterización en frecuencia de la señal, entre otras. Una opción bastante habitual es que estas características se deriven de una segmentación del ECG, similar a la realizada por el experto humano. El objetivo de dicha segmentación es identificar las ondas constituyentes del latido: P, Q, R, S y T, y medir su amplitud y anchura, así como la distancia entre complejos, o medidas derivadas [78]. Esta representación en términos de ondas se corresponde con los principales procesos que suceden en el miocardio: las despolarizaciones y repolarizaciones de aurículas y ventrículos. Una ventaja adicional de utilizar estas características para representar al latido es que realiza una abstracción sobre la señal en los mismos términos (ondas, segmentos, latidos) que el experto utiliza para interpretar el ECG (véase Figura 1.8) [31][60]. Sin embargo, hasta la fecha el problema de la delineación del latido (identificar sus ondas constituyentes) no está resuelto de forma completamente satisfactoria. Ningún método de la bibliografía ofrece garantías de fiabilidad y robustez para obtener una delineación suficientemente estable y precisa en registros ambulatorios, que suelen contener abundante ruido. Esto puede causar que al analizar y comparar latidos se encuentren diferencias derivadas de imprecisiones en la representación del latido, y no de un origen distinto o camino de propagación diferente.

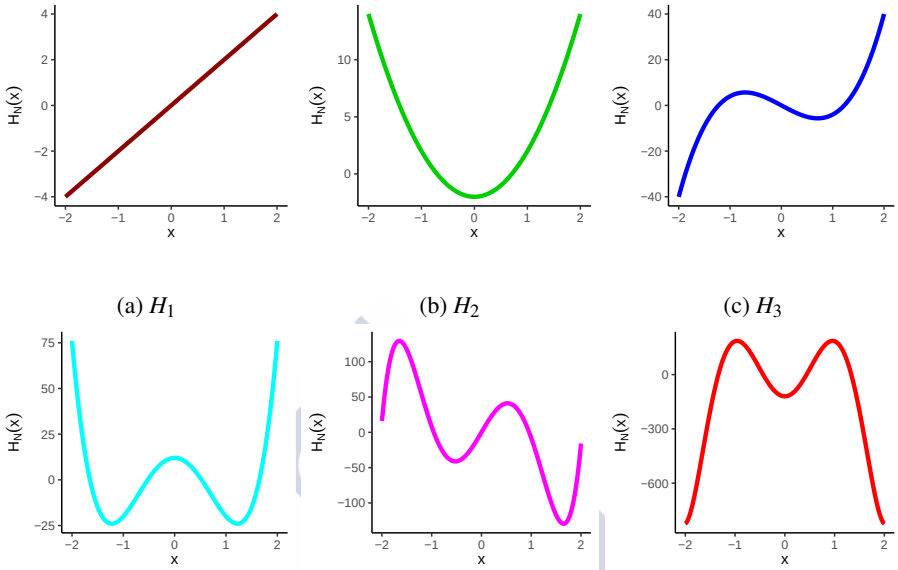
3. **Bases de funciones**. Se utiliza una base de funciones para modelar el latido, que se representa como una combinación de dichas funciones. Esta representación del ECG mediante bases de funciones aparece por primera vez en una propuesta basada en los polinomios ortogonales de Laguerre [149]. Posteriormente se aplicaría la expansión de Karhunen-Loeve, que proporciona una representación óptima respecto al error cuadrático medio [3], y la transformada wavelet, que permite una representación en múltiples niveles de resolución [121]. Recientemente se han propuesto las funciones ortogonales de Hermite (véase Figura 2.1), que incluyen un parámetro de anchura [79]. En todos estos casos, el latido es representado por un vector con los coeficientes que permiten reconstruirlo a partir de la combinación de las funciones base. La ventaja de esta representación es un mejor comportamiento frente al ruido y los artefactos [66][77]; además de ser una representación compacta. En su contra, se pierde el significado fisiológico de los criterios de clasificación, lo que disminuye la interpretabilidad de los resultados e impide su uso si las prestaciones no son excepcionales.

La bibliografía recoge un amplio conjunto de propuestas de representación del latido que pueden encuadrarse bajo alguno de los tres esquemas generales de representación aquí descritos, o bajo una combinación de ellos. Ninguna representación ha demostrado ser inherentemente mejor que el resto, eligiendo los autores una representación u otra de acuerdo al propósito del trabajo o a su conocimiento previo.

La representación a partir de bases de funciones es una de las alternativas preferidas de la bibliografía [3][70][105][116]. El conjunto de funciones más comúnmente utilizado para representar latidos es el compuesto por las funciones de Hermite. Una búsqueda en Google Scholar con los términos “Hermite representation electrocardiogram” devuelve cerca de 2800 artículos científicos, de los cuales más de 1000 han sido escritos desde el año 2012. Parte de esta popularidad se debe a la similitud que guardan con el complejo QRS [77][130], que también tiene un cierto grado de simetría (especialmente la onda R). Otro punto fuerte de estas funciones es que incluyen un parámetro de anchura o dilatación que permite representar de forma eficiente latidos con diferentes anchuras del complejo QRS.

El polinomio de Hermite $H_n(x)$ se puede obtener de forma recursiva:

$$H_n(x) = 2xH_{n-1}(x) - 2(n-1)H_{n-2}(x), \tag{2.1}$$



(a) H_1 (b) H_2 (c) H_3

(d) H_4 (e) H_5 (f) H_6

Figura 2.1: Forma de las primeras funciones de Hermite.

donde $H_0(x) = 1$ y $H_1(x) = 2x$. Por ejemplo, $H_2(x) = 4x^2 - 2$, $H_3(x) = 8x^3 - 12x$ y así sucesivamente.

Al utilizar Hermite, o cualquier otra base de funciones, para representar los latidos debemos realizar una elección sobre cuántos coeficientes, es decir cuántas funciones, utilizaremos para la representación del latido. Como regla general, cuantas más funciones utilicemos más exacta será la representación del latido. Pero al mismo tiempo un número alto de funciones implica una alta dimensionalidad del espacio de características, algo no deseable. Además, emplear un número muy elevado de funciones puede llevar a representar elementos del ECG que no forman parte realmente de la morfología del latido, como por ejemplo ruido o artefactos de la señal.

En los múltiples trabajos que utilizan las funciones de Hermite para representar latidos no se aprecia un consenso en el número de funciones a utilizar. Hay trabajos que utilizan tan solo 3 funciones [16], mientras que otros usan hasta 20 [112], pasando por trabajos que usan otras cantidades intermedias: 6 [77], 11 [57] ó 15 [146]. Los autores de estos trabajos

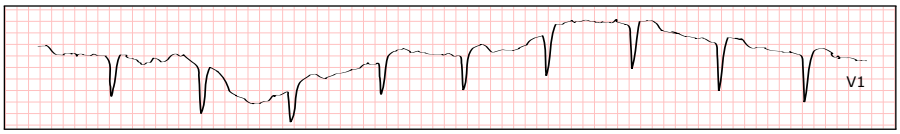


Figura 2.2: Fragmento de un ECG en el que se puede observar cómo afecta el ruido de la deriva de línea base. (Fuente: MIT-BIH Arrhythmia Database, registro 108, entre 0:00:53 y 0:01:03)

generalmente no justifican el número de funciones elegido más allá del uso de una simple inspección visual de la reconstrucción de unos pocos latidos. El uso de un modelo matemático para la representación del latido cardiaco permite emplear técnicas de teoría de la información (criterios como BIC [124] o AIC [5]) a la hora de evaluar la calidad de dicha representación. Sin embargo, hasta la fecha ningún otro autor ha empleado este tipo de técnicas para evaluar cuál es el número óptimo de funciones para la representación del latido cardiaco. Este será el propósito de este capítulo. Es conveniente indicar que el análisis y discusión aquí realizados van más allá de la utilización de las funciones de Hermite para el agrupamiento, objeto de la tesis, queriendo ser de utilidad para cualquier forma de procesamiento del ECG que se plantee la representación del complejo QRS como punto de partida.

2.1. Material y métodos

2.1.1. Preprocesado de los registros de electrocardiograma

El ruido y su consecuente intento de filtrado es una constante en el análisis de cualquier señal biomédica. Para el ECG, siendo una de las señales más estudiadas y caracterizadas, existe una extensa bibliografía para el tratamiento del ruido [136][140]. La máxima componente frecuencial del latido en el ECG está en el complejo QRS, con un rango entre los 4 y 20 Hz [135]. Todas las componentes frecuenciales por encima de dicho rango deben ser tratadas como ruido. Entre las más importantes encontramos el ruido de la red eléctrica, en torno a los 50-60 Hz, o el ruido que se genera en la interfase de los electrodos, en torno a los 60 Hz [1]. Basándonos en esta información diseñamos un filtro Butterworth paso bajo con una frecuencia de corte de 40 Hz para eliminar el ruido de alta frecuencia. De esta forma damos un margen suficiente para no eliminar componentes de la señal, y al mismo tiempo eliminamos la mayor parte del ruido de alta frecuencia. En nuestro caso es preferible no ser

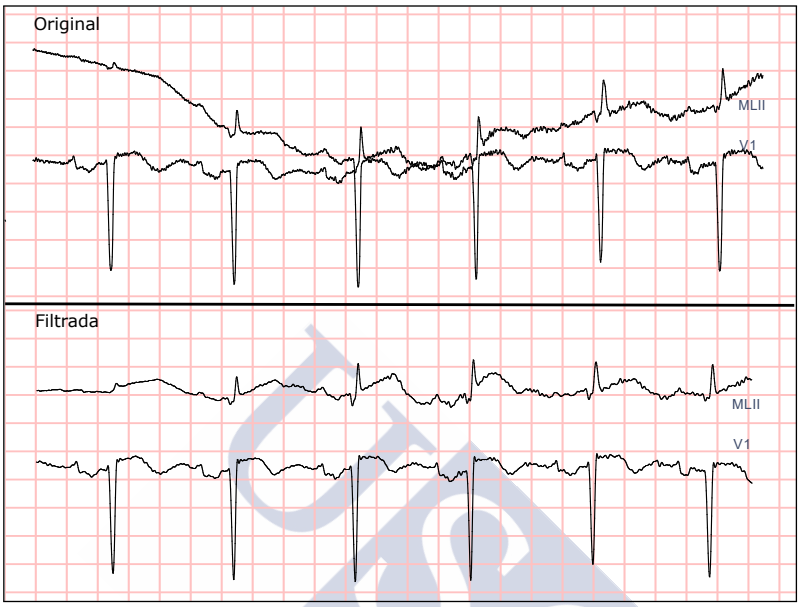


Figura 2.3: Extracto de señal con deriva en línea base y ruido de alta frecuencia antes y después del filtrado. (Fuente: MIT-BIH Arrhythmia Database, registro 208, entre 0:05:17 y 0:05:22)

demasiado agresivos con el filtrado de alta frecuencia ya que la representación con funciones de Hermite del latido es robusta frente al ruido.

La eliminación de la deriva de línea base es más complicada que la del ruido de alta frecuencia (véase Figura 2.2). Su componente frecuencial es más baja, habitualmente en el rango de los 0-0.8 Hz, y se solapa con componentes de frecuencia de características del latido que sí queremos conservar: las ondas P o T tienen una componente frecuencial en el rango de los 0.5-10 Hz. En nuestro caso optamos por utilizar filtros de fase mínima Daubechies [14]. Utilizamos la transformada wavelet para reconstruir una aproximación del espectro de la señal por debajo de 1 Hz. Necesitaremos calcular hasta qué nivel debemos realizar la reconstrucción en base a la frecuencia de muestreo; por ejemplo para 360 Hz necesitaremos llegar hasta el nivel 8 (inclusive) de la transformada. Posteriormente esta reconstrucción de la señal es restada a la señal original, eliminando por tanto la deriva en línea base [141][150]. En la Figura 2.3 se puede apreciar un ejemplo del filtrado realizado.

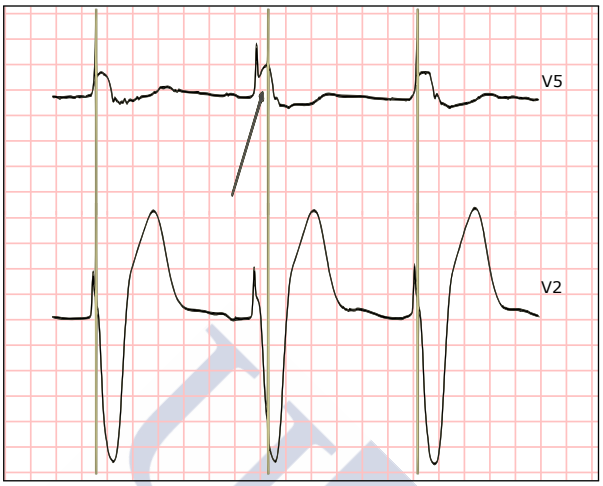


Figura 2.4: Ejemplo de las anotaciones realizadas por los cardiólogos; obsérvese la variación de la posición de las anotaciones de los latidos. (Fuente: MIT-BIH Arrhythmia Database, registro 102, entre 0:00:29 y 0:00:33)

2.1.2. Búsqueda del punto de máxima simetría en el latido

Se utilizará la base de datos MIT-BIH Arrhythmia Database y se partirá de las anotaciones de la base de datos que sitúan cada latido en un instante de tiempo. De esta forma se facilita la comparación de los resultados con otros trabajos de la bibliografía.

Las funciones de Hermite son simétricas en torno a su eje central. Por tanto, al menos teóricamente, para obtener una mejor representación utilizando funciones de Hermite deberíamos hacer coincidir este eje central con el instante de máxima simetría del latido. Este punto es, generalmente, el pico máximo del complejo QRS, la onda R. Sin embargo, las anotaciones realizadas por los cardiólogos, o por algoritmos automáticos, no señalan siempre ese punto del latido (véase Figura 2.4). Establecer el centro del latido en una posición estable y de máxima simetría debería mejorar los resultados de la representación basada en funciones de Hermite. Por ello aplicaremos un algoritmo de corrección de la posición del latido, cuyo propósito es tratar de acercar la anotación del latido a la posición de máxima simetría.

En primer lugar, para la derivación seleccionada, se calcula la media de la señal en una ventana de 200 ms alrededor de la anotación original. Posteriormente buscaremos el punto

más alejado del valor de la media dentro de dicha ventana, que generalmente será el pico de la onda R. Esta será la nueva posición de la anotación del latido.

El algoritmo de corrección de las anotaciones se puede aplicar en una derivación del ECG y emplear las posiciones obtenidas para las restantes, o puede aplicarse sobre cada derivación por separado. Si se aplica únicamente a una derivación, se asume que la posición del pico de la onda R es igual en el resto de derivaciones (lo que no es necesariamente cierto en la práctica). De aplicarse el algoritmo a cada derivación, la localización en el tiempo del latido puede ser ligeramente distinta para cada derivación. En este capítulo probaremos tres estrategias: usar las anotaciones originales de la base de datos; corregir la posición sobre una única derivación y usar dicha posición sobre el resto de derivaciones; y corregir la posición sobre cada derivación independientemente.

2.1.3. Cálculo de la representación de Hermite

El complejo QRS es la parte central del latido cardiaco y su característica más importante. Para el problema que nos ocupa, el del agrupamiento de latidos, la onda T proporciona poca información adicional si ya tenemos en cuenta la proporcionada por el complejo QRS. La onda P sí que proporciona información adicional que ayuda a distinguir entre distintas arritmias, como las contracciones auricular y auriculoventricular prematuras, y los latidos de escape auriculares y auriculoventriculares. El problema en este caso reside en identificar y extraer la onda P de forma precisa y estable [34]. Este hecho lleva a que habitualmente se intente obtener información equivalente de otra forma más robusta, siendo la opción más habitual emplear información derivada de medir la distancia entre latidos consecutivos [30][77][139]. Por tanto, en esta sección nos centraremos en la representación del complejo QRS mediante las funciones de Hermite, asumiendo que esta representación se complementará con información extraída de la distancia entre latidos.

Para el cómputo de la representación se parte de las anotaciones que sitúan el latido en un instante de tiempo. Alrededor de estas anotaciones se extrae una ventana de 200 ms para cada complejo. El tamaño de esta ventana es suficientemente grande para contener la anchura del complejo QRS de un latido ventricular, pero al mismo tiempo suficientemente estrecha como para dejar fuera las ondas P y T. Este valor para la anchura de esta ventana es uno de los más comunes en la bibliografía [77].

Las funciones de Hermite convergen a cero en $\pm\infty$. Para asegurar esta convergencia se añaden 100 ms de ceros en ambos extremos de la ventana de 200 ms que contiene el complejo

QRS. Se obtiene por tanto una ventana de 400 ms, $x[l]$, representada como:

$$\begin{aligned} x[l] &= \sum_{n=0}^{N-1} c_n(\sigma)\phi_n[l,\sigma) + e[l], \\ l &= \left\{ -\left\lfloor \frac{W \cdot f_s}{2} \right\rfloor, -\left\lfloor \frac{W \cdot f_s}{2} \right\rfloor + 1, \ldots, \left\lfloor \frac{W \cdot f_s}{2} \right\rfloor \right\}, \end{aligned} \tag{2.2}$$

siendo N el número de funciones de Hermite utilizadas, W el tamaño de la ventana en segundos y f_s la frecuencia de muestreo de la señal. Los símbolos $\lfloor\ \rfloor$ significan que el valor es redondeado al entero menor más cercano. $\phi_n[l,\sigma)$ es la n función discreta de Hermite obtenida discretizando la función continua $\phi_n(t,\sigma)$ a una frecuencia f_s; c_n son los coeficientes de la combinación lineal que representa nuestro latido; $e[l]$ es el error entre $x[l]$ y la representación de Hermite; y σ es un parámetro de elongación que controla la anchura de la función de Hermite, permitiendo ajustarla a la anchura del complejo QRS.

Las funciones de Hermite $\phi_n[l,\sigma)$, $0 \leq n < N$, son definidas como:

$$\phi_n[l,\sigma) = \frac{1}{\sqrt{\sigma 2^n n! \sqrt{\pi}}} e^{-(l \cdot T_s)^2/2\sigma^2} H_n(l \cdot T_s/\sigma) \tag{2.3}$$

siendo T_s el inverso de la frecuencia de muestreo.

De esta forma, cada complejo QRS se representa mediante los N coeficientes $c_n(\sigma)$, $0 \leq n < N$, y el valor de σ. Podemos ver en la Figura 2.5 cómo varía la representación utilizando distintos valores de N. Generalmente, usar más funciones implica una representación más precisa. Sin embargo, utilizar un mayor número de funciones también conlleva el riesgo de sobreajustar y modelar el ruido de la señal y no la forma real del complejo QRS. En la misma figura se puede ver este comportamiento cuando se utilizan 15 funciones.

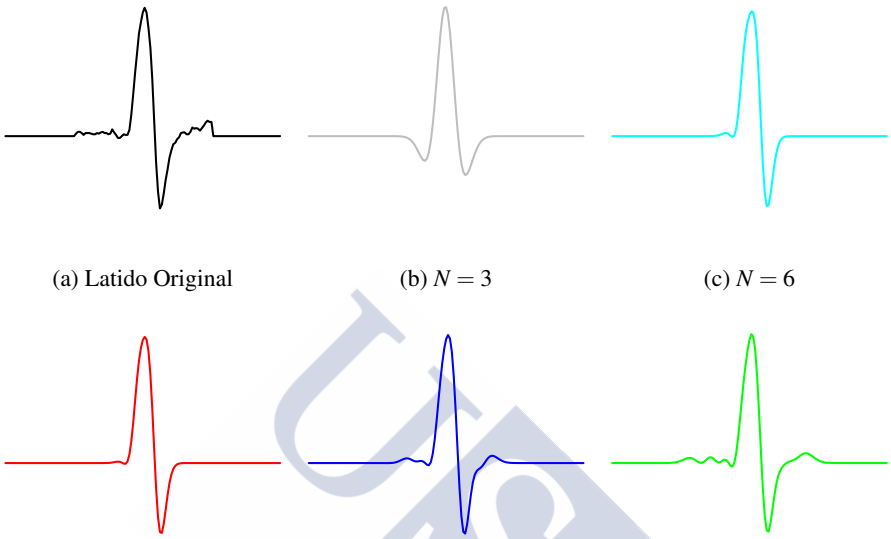
Para un cierto valor de σ, las funciones de Hermite forman una base ortonormal:

$$\int_{-\infty}^{\infty} \phi_n(\sigma)\phi_m(\sigma) = \delta_{mn}. \tag{2.4}$$

Esto permite un cálculo eficiente de $c_n(\sigma)$. Sin una ventana infinita, o en el caso de funciones discretas, (2.4) no se cumple. Sin embargo, si $\phi_n[l,\sigma)$ es suficientemente cercano a cero en los extremos y fuera de la ventana, (2.4) resulta una aproximación aceptable.

Para los límites de la ventana usaremos el criterio (bastante tolerante) de que $\phi_n[l,\sigma)$ sea como mucho $1/10$ de su máximo valor dentro de la ventana:

$$|\phi_n[-l_0,\sigma)| = |\phi_n[l_0,\sigma)| < \frac{1}{10} \max_{l \in [-l_0, l_0]} |\phi_n[l,\sigma)|, \tag{2.5}$$



(a) Latido Original (b) $N = 3$ (c) $N = 6$

(d) $N = 9$ (e) $N = 12$ (f) $N = 15$

Figura 2.5: Latido original y representación de Hermite con N=3, 6 ,9 ,12 y 15 para un mismo valor de σ.

donde $-l_0$ y l_0 son respectivamente la primera y última muestra de la ventana. Requeriremos también que el valor de $\phi_n[l,\sigma)$ fuera de la ventana sea menor que en el límite de la ventana:

$$|\phi_n[l,\sigma)| \leq |\phi_n[l_0,\sigma)| \ \ \forall |l| > l_0. \tag{2.6}$$

Para un determinado tamaño de ventana y un número fijo de funciones de Hermite, (2.5) y (2.6) imponen un límite máximo en el valor de σ. Por ejemplo, para $N = 3$, 4 y 5, y una ventada de 200 ms de señal, los valores máximos de σ que cumplen (2.5) y (2.6) son 62 ms, 55 ms y 51 ms, respectivamente.

Una vez fijado el valor de σ podemos calcular los coeficientes $c_n(\sigma)$ minimizando la suma cuadrática del error:

$$\sum_l (e[l])^2 = \sum_l \left(x[l] - \sum_{n=0}^{N-1} c_n(\sigma)\phi_n[l,\sigma) \right)^2. \tag{2.7}$$

El mínimo de este error cuadrático podemos aproximarlo a partir de la propiedad de ortogonalidad si suponemos que:

$$c_n(\sigma) = \vec{x} \cdot \vec{\phi}_n(\sigma), \tag{2.8}$$

dónde los vectores, $\vec{x}$ y $\vec{\phi}_n(\sigma)$, son definidos como $\vec{x} = \{x[l]\}$ y $\vec{\phi}_n(\sigma) = \{\phi_n[l,\sigma)\}$.

Por tanto, podemos obtener el mejor σ mediante un proceso iterativo [77]. Para ello se realizará un incremento iterativo de σ recalculando para cada iteración (2.8) y (2.7). En este proceso σ comienza desde 0 y se va incrementando hasta el valor máximo (fijado por (2.5) y (2.6)) en incrementos de $\frac{f_s}{1000}$. Finalmente seleccionaremos el valor que minimiza la suma cuadrática del error. Los valores óptimos de σ para cualquier N en la base de datos MIT-BIH Arrhythmia Database están mayormente en un rango entre 14 y 21 ms, bastante por debajo que el límite máximo marcado.

Terminado este paso ya tenemos los valores que conformarán la representación del complejo QRS: el valor de σ y los valores de los coeficientes de las funciones de Hermite $c_0(\sigma), c_1(\sigma), c_2(\sigma), ..., c_{N-1}(\sigma)$. Si en nuestro problema se van a considerar varias derivaciones en el electrocardiograma, cada derivación será procesada independientemente, obteniendo una representación para cada complejo QRS.

2.1.4. Error en la representación

Una vez obtenida la representación del complejo QRS es necesario estudiar el error entre la representación y la señal original. Para ello existen múltiples métricas muy variadas. En [77] para calcular el error en la representación de Hermite se utiliza el cociente entre la energía del error cuadrático (2.7), y la energía de la señal original:

$$Err = \frac{\sum_l (e[l])^2}{\sum_l (x[l])^2}. \tag{2.9}$$

Esta es una métrica del error directa y sencilla pero su valor resulta de difícil interpretación. No obstante, nos servirá para comparar nuestros resultados con los obtenidos en dicho trabajo. Además, calcularemos la raíz del error cuadrático medio normalizada (NRMSD):

$$NRMSD = \frac{RMSD}{x_{max} - x_{min}} = \frac{\sqrt{\frac{\sum_t (e[l])^2}{V}}}{x_{max} - x_{min}}, \tag{2.10}$$

donde V es el tamaño de la ventana en muestras y x_{min} y x_{max} son, respectivamente, el valor mínimo y máximo de la señal en dicha ventana. Esta medida tiene una interpretación más

intuitiva que (2.9), siendo el error promedio expresado como un porcentaje del rango de valores de la señal.

2.1.5. Selección de la representación óptima del complejo QRS

Añadir más funciones a la representación del latido siempre reduce el error, obteniendo por tanto una representación más exacta, pero a costa de incrementar la dimensionalidad del vector de características que representa el latido y los requerimientos computacionales para su procesamiento. En la bibliografía no hay ningún criterio bien definido para decidir cuál es el número adecuado de funciones a utilizar en la representación, sino que generalmente esta tarea se lleva a cabo mediante una inspección visual del resultado de la reconstrucción de unos pocos latidos. Dado que la representación se basa en un modelo matemático, en esta sección aplicaremos técnicas de la literatura de selección de modelos [17][18][22] que proporcionan un criterio objetivo para la selección del número óptimo de funciones para la representación del latido.

En el campo de las técnicas de selección de modelos encontramos principalmente dos aproximaciones: criterios bayesianos y criterios basados en la teoría de la información. Entre las aproximaciones basadas en criterios bayesianos destaca el criterio de información bayesiana (Bayesian Information Criterion, BIC) [124]. Las aproximaciones basadas en teoría de la información son muchas y variadas, pero podemos destacar por su extendido uso el criterio de información de Akaike (Akaike Information Criterion, AIC) [5].

BIC está basado en la función de verosimilitud:

$$L = p(x \mid \hat{\theta}, M), \tag{2.11}$$

donde $\hat{\theta}$ son los parámetros, x los datos observados y M el modelo. Al añadir parámetros adicionales la verosimilitud siempre tiende a aumentar. Sin embargo, si dejamos que la verosimilitud crezca sin límites estaríamos sobreajustando el modelo. Para resolver esta limitación, BIC añade un término de penalización basado en el número de parámetros del modelo. En el caso que nos atañe cada nuevo parámetro es un nuevo coeficiente, resultado de añadir una función de Hermite a la representación. Por ejemplo, si utilizamos funciones hasta $\phi_{N-1}[l,\sigma)$ tendremos $N+1$ parámetros, los N coeficientes $c_0(\sigma), c_1(\sigma), c_2(\sigma), ..., c_{N-1}(\sigma)$ de la combinación lineal y σ. Por tanto, para nuestro problema BIC se formula como:

$$\text{BIC} = -2 \cdot \ln \hat{L} + k \cdot \ln(m), \tag{2.12}$$

dónde m es el número de muestras, k es el número de parámetros ($k = N + 1$ cuando usamos N funciones) y $\hat{L}$ es el valor máximo de la función de verosimilitud.

AIC está basado en la distancia de Kullback-Leibler y en la entropía de la información. Mediante su uso podemos obtener una estimación de la distancia relativa entre un modelo y el proceso desconocido que verdaderamente generó los datos observados. En la práctica se usa para comparar distintos modelos y seleccionar el que presente la menor distancia, pero no permite saber nada acerca de la calidad del modelo en sentido absoluto. Si todos los modelos candidatos se adaptan pobremente a los datos, AIC no lo detectará. Akaike en su trabajo definió AIC como:

$$\mathrm{AIC} = -2 \cdot \ln \hat{L} + 2 \cdot k. \tag{2.13}$$

Esta formulación clásica de AIC es poco adecuada para los casos en que el tamaño muestral (m) no sea varios órdenes de magnitud mayor que el número de parámetros (k), ya que tiende a sobreajustar los datos. Sugiura [133] propuso una variante de segundo orden, AIC_c, con un término de corrección adicional para paliar este efecto. En el caso de que m sea suficientemente grande AIC_c converge a AIC y por tanto seleccionarán el mismo modelo. Esta corrección AIC_c se define como:

$$\mathrm{AIC}_c = \mathrm{AIC} + \frac{2k(k+1)}{m-k-1}. \tag{2.14}$$

Dado que en nuestro problema el cociente m/k no es grande, haremos uso de esta variante de AIC.

Tanto AIC como BIC tienen sus ventajas e inconvenientes. AIC tiende a elegir modelos más complicados de lo estrictamente necesario, especialmente si el número de muestras es pequeño [17]. Pero al estar basado en una minimización del error cuadrático medio se convierte en un candidato idóneo para un problema como el que nos atañe. Por otra parte, BIC debería, en teoría, ser aplicado únicamente para encontrar el modelo real entre un conjunto de modelos candidatos que lo contiene; condición que no se cumple en nuestro caso. Además, BIC muestra una tendencia a seleccionar modelos demasiado simples si el número de muestras es pequeño, justo al contrario que AIC. Aprovechando que ambos criterios abordan el mismo problema desde distintas aproximaciones, aplicaremos tanto AIC como BIC por separado para luego comparar sus resultados [148].

Bajo las hipótesis de normalidad e independencia de los errores, y asumiendo además que la varianza es constante, BIC y AIC_c pueden ser expresados como [17]:

$$\mathrm{BIC} = m \cdot \ln(\sigma_e^2) + k \cdot \ln(m), \tag{2.15}$$

$$\mathrm{AIC}_c = m \cdot \ln(\sigma_e^2) + 2 \cdot k + \frac{2k(k+1)}{m-k-1}, \tag{2.16}$$

dónde σ_e^2 es la varianza del error. El estimador insesgado $\widehat{\sigma_e^2}$ será usado para estimar σ_e^2, que en nuestro caso adopta la siguiente forma:

$$\widehat{\sigma_e^2} = \frac{1}{m-1} \sum_{l=1}^{m} \left(x[l] - \sum_{n=0}^{N-1} c_n(\sigma)\phi_n[l,\sigma) \right)^2 = \frac{1}{m-1} \sum_{l=1}^{m} e[l]^2. \tag{2.17}$$

Cada complejo QRS de cada derivación se considera un problema de selección de modelos independiente. Para cada uno de ellos calcularemos BIC y AIC_c usando desde 2 hasta 30 funciones de Hermite para construir la representación. Posteriormente, empleando cada uno de los dos criterios se selecciona el número óptimo de funciones de Hermite a utilizar.

2.1.6. Selección de características

Cada complejo QRS es representado por los coeficientes de la combinación lineal de funciones $c_n(\sigma)$ y por el parámetro de anchura σ. Si tenemos en cuenta que generalmente trabajaremos al menos con dos derivaciones del ECG y que podemos llegar a usar hasta 30 funciones para representar el complejo QRS, podríamos tener hasta 62 características para representar cada latido, o incluso más en el caso de utilizar un mayor número de derivaciones (362 para 12 derivaciones en la INCARTDB). En este contexto, podría tener sentido intentar seleccionar un subconjunto de aquellas características que son más relevantes para separar unas familias morfológicas de latidos de otras.

Con este fin se aplicaron varias técnicas de selección de características que permiten ordenar las características según la cantidad de información que aportan. Las técnicas utilizadas fueron dos basadas en la información mutua "InformationGain" y "GainRatio" y una basada en el test Chi-cuadrado. Las dos primeras tratan de medir la información que aporta cada nueva característica para tomar decisiones (*InformationGain*) [115], aplicando *GainRatio*, una corrección sobre la anterior para penalizar decisiones que den lugar a un gran número de bifurcaciones en la decisión. Por otra parte, el método basado en Chi-cuadrado ordena las características basadas en el valor del estadístico chi-cuadrado con respecto a la clase [91].

Para aplicar estas técnicas se utilizó la base de datos MIT-BIH al completo. La selección de características fue aplicada para cada registro (paciente) de forma independiente y sobre

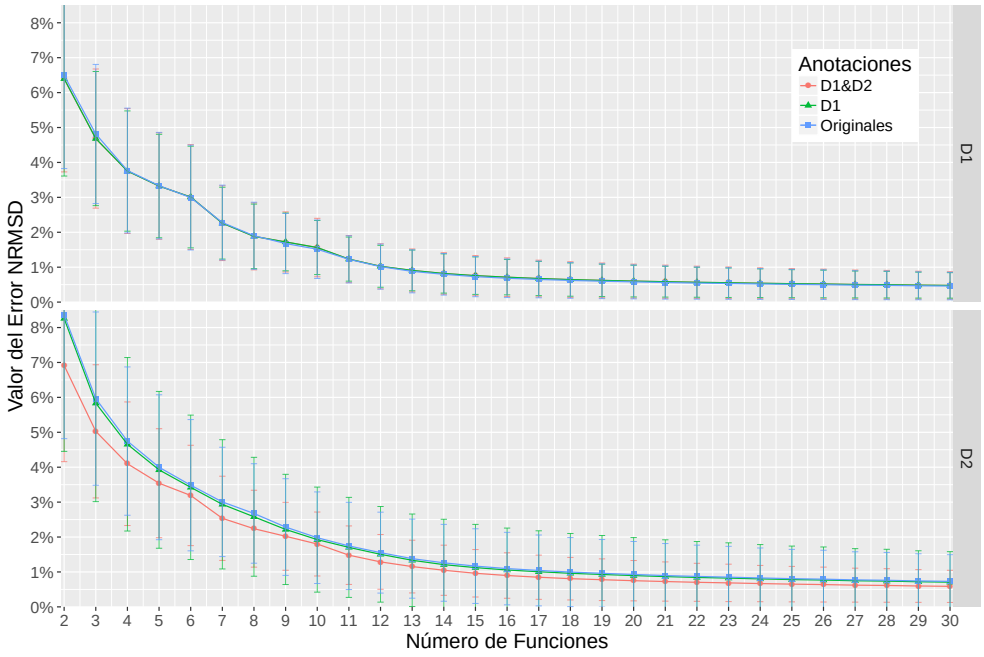


Figura 2.6: Resultados del error NRMSD por derivación para la señal filtrada para las tres estrategias de corrección de posición del latido: sin corrección (Originales), corrección sobre la primera derivación (D1) y corrección sobre las dos derivaciones (D1&D2).

toda la base de datos en su conjunto. De esta forma podemos comparar los resultados de ambas aproximaciones.

2.2. Resultados

Los algoritmos descritos en este capítulo fueron implementados en el lenguaje Java, con la excepción del filtrado de línea base y el filtrado de alta frecuencia. Dichos filtros fueron implementados en Matlab y desde ese entorno se generó un conjunto completo de los registros de la base de datos MIT-BIH filtrados. Posteriormente se procesaron tanto los registros filtrados como los registros originales con los algoritmos implementados en Java.

2.2.1. Error en la representación

Los resultados del error NRMSD (2.10) se muestran en las Figuras 2.6 y 2.7. En dichas figuras se muestra el error de cada derivación; las barras muestran la desviación estándar de dicho error. En la Figura 2.6 se muestran los resultados con la señal filtrada mientras que en la Figura 2.7 se muestran con la señal sin filtrar; correspondiendo las gráficas de la parte superior a la primera derivación y las de la parte inferior a la segunda derivación en ambos casos. Para cada caso aparecen tres valores de error correspondientes con cada una de las tres estrategias de corrección de posición de latido recogidas en la Sección 2.1.2: utilizar las anotaciones originales de la base de datos; corregir la posición solo sobre la primera derivación (D1); y corregir la posición sobre ambas derivaciones por separado (D1&D2). Se eligió la primera derivación al corregir las anotaciones sobre la posición de una única derivación debido a que originalmente fue la derivación utilizada para anotar la base de datos MIT-BIH. Estas tres aproximaciones se muestran en las figuras en distintas series de datos utilizando cuadrados, triángulos y círculos respectivamente.

En otra gráfica (véase Figura 2.8) se muestran los resultados del error calculado según la ecuación (2.9); el error mostrado es la media del error en cada una de las dos derivaciones. En este caso se muestra en la misma figura los resultados con la señal filtrada (arriba) y con la señal sin filtrar (abajo) utilizando las mismas tres estrategias de corrección de posiciones de los latidos que en las figuras anteriores.

2.2.2. Representación óptima según AIC y BIC

En la Figura 2.9 podemos ver el porcentaje de latidos de la base de datos MIT-BIH cuyos complejos QRS son óptimamente representados, de acuerdo a BIC y AIC_c, con N o menos funciones de Hermite. Por ejemplo, el 57 % de los complejos son representados óptimamente con 20 o menos funciones de Hermite, de acuerdo tanto a BIC como a AIC_c. Ningún complejo QRS se representó óptimamente con un número menor de 8 funciones; algunos complejos fueron representados óptimamente utilizando 9 o 10 funciones, pero su número es muy bajo y por ello no son mostrados en la gráfica. Los resultados mostrados fueron calculados utilizando la señal filtrada, siendo muy similares a los resultados obtenidos sin filtrar.

Es necesario especificar que de los latidos que aparecen como representados óptimamente con 30 funciones en la Figura 2.9 pueden en algunos casos requerir un mayor número de funciones para representarse de modo óptimo, por lo que se podría decir que el análisis se

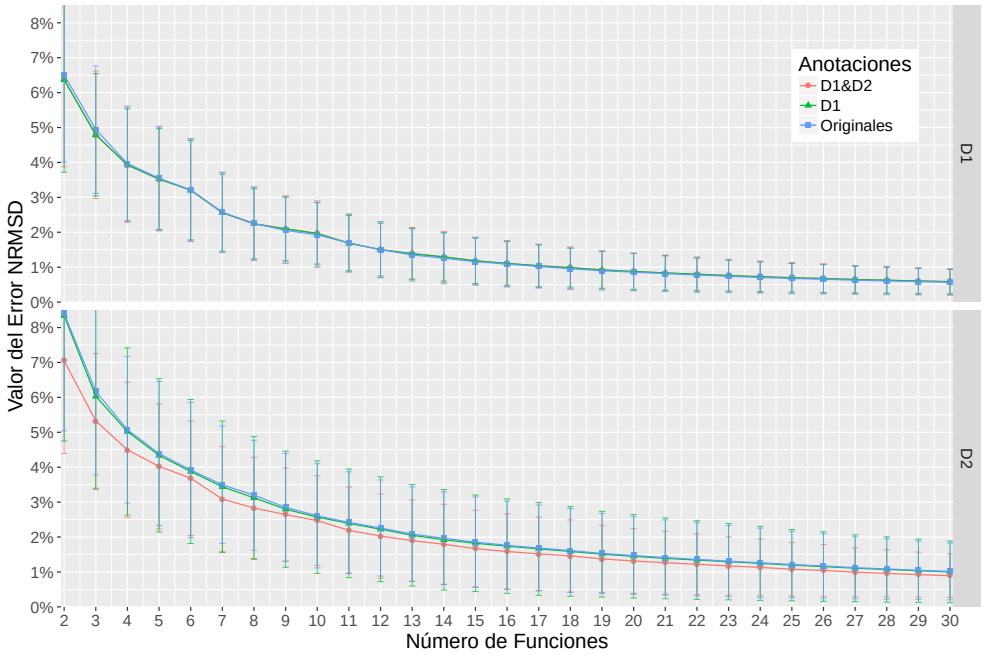


Figura 2.7: Resultados del error NRMSD por derivación para la señal sin filtrar para las tres estrategias de corrección de posición del latido: sin corrección (Originales), corrección sobre la primera derivación (D1) y corrección sobre las dos derivaciones (D1&D2).

realizó únicamente desde 2 hasta 29 funciones, aunque se calcularan también los valores para 30 funciones. Por tanto, para aquellos latidos que hubieran requerido un mayor número de funciones AIC_c y BIC seleccionarán la representación con 30 funciones como el modelo óptimo. Sin embargo, el 97 % de los latidos según BIC y el 99 % de los latidos según AIC_c están ya óptimamente representados con 29 o menos funciones. Por tanto, el número de latidos que se representarían óptimamente con más de 30 funciones sería, sin duda, pequeño.

2.2.3. Selección de características

Los resultados obtenidos por los métodos de selección de características no fueron alentadores. Las características más relevantes que fueron seleccionadas por los tres métodos variaban considerablemente entre pacientes. Por otra parte, las características seleccionadas para cada paciente eran distintas a las seleccionadas trabajando con toda la base de datos

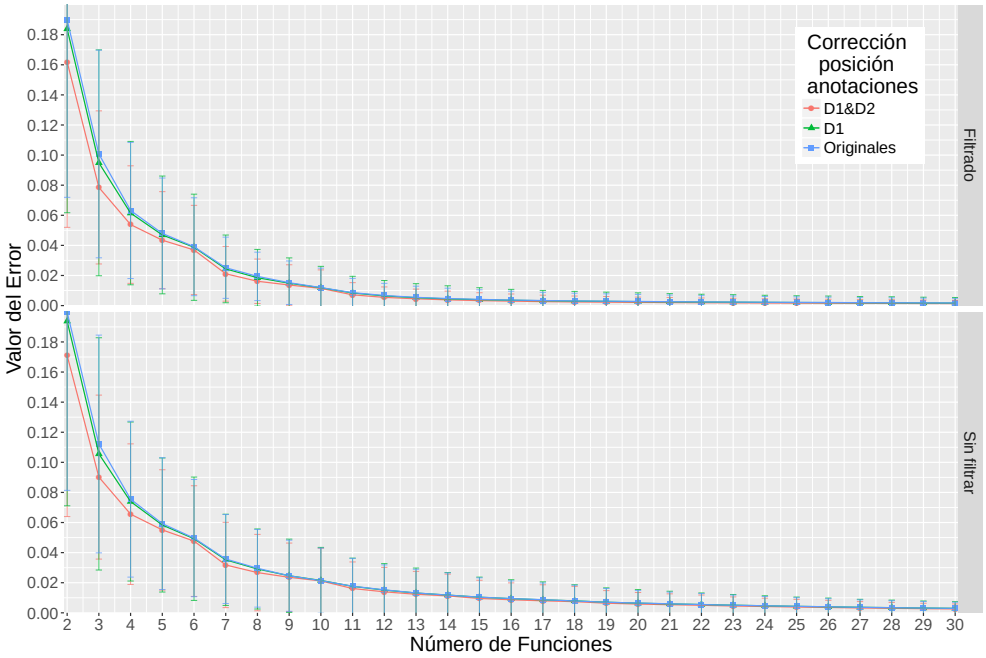


Figura 2.8: Resultados de error utilizando la medida de error de la ecuación (2.9).

como conjunto. Tras muchos intentos, se consideró necesario desestimar la posibilidad de que estas técnicas proporcionaran información útil aplicable a todos los pacientes. Los resultados que se obtuvieron sugieren que es posible reducir el espacio de características para un determinado paciente, pero la estrategia en la que se debe basar esa reducción es específica para cada paciente. Las características que para un paciente no son relevantes, pueden ser relevantes para otro y viceversa. Por tanto, la selección de características tendría que ser aplicada paciente a paciente, conclusiones similares a las ya obtenidas por [120]. Dado que nuestro objetivo final es construir un agrupador automático de latidos, aplicable a todos los pacientes, no podemos asumir que tenemos un subconjunto de latidos anotados que pudieran ser utilizados en una etapa de selección de características específica para cada paciente. Por ello, trabajaremos con el conjunto de características completo.

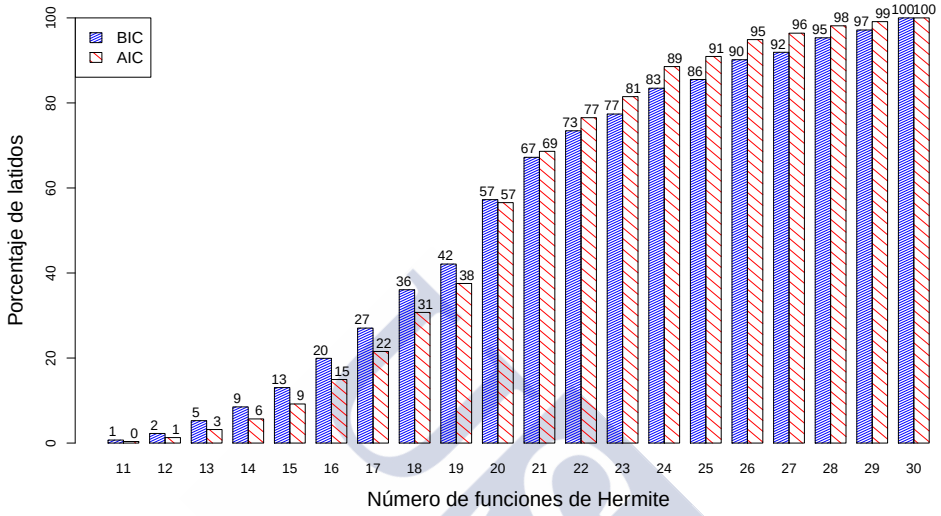


Figura 2.9: Porcentaje de complejos QRS que pueden ser óptimamente representados con N o menos funciones de Hermite.

2.3. Coste computacional de la representación de Hermite

En los test ejecutados se midió el coste computacional requerido para calcular la representación de Hermite de cada latido. Dicha representación fue calculada utilizando un AMD Opteron 6200 series a 1.6 GHz con 16 núcleos y 128 GB de RAM corriendo en Linux (Debian 3.2). Para acelerar el cálculo de la representación se paralelizó el cálculo de la representación utilizando 16 hilos concurrentes (uno para cada núcleo), permitiendo calcular la representación de hasta 16 latidos al mismo tiempo.

Los resultados del tiempo de ejecución por cada latido son mostrados en la Figura 2.10, utilizando desde 2 hasta 30 funciones. Las diferencias en el tiempo de ejecución entre las distintas estrategias de corrección de posición de las de anotaciones y entre la señal filtrada y sin filtrar son inapreciables por lo que se muestra únicamente una serie de datos.

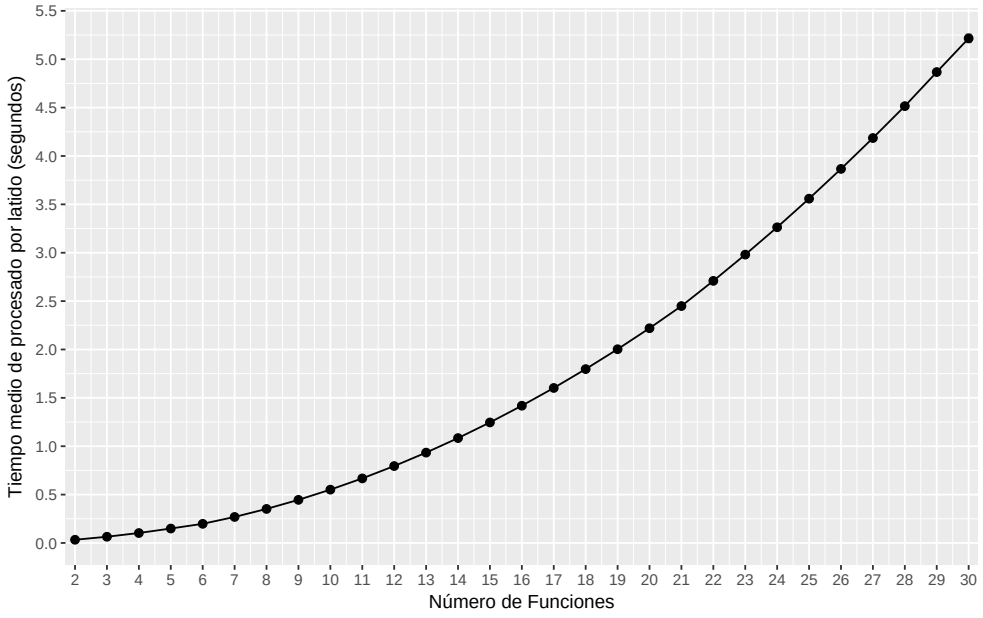


Figura 2.10: Tiempo medio de ejecución necesario para calcular la representación de Hermite con distinto número de funciones.

2.4. Discusión

Los resultados del cálculo del NRMSD y de la medida de error dada por la Ecuación (2.9) mostrados en las Figuras 2.6, 2.7 y 2.8 muestran que incluso con un número pequeño de funciones de Hermite podemos representar los latidos de forma aceptable (error NRMSD menor del 8 %). Esto no es sorprendente ya que hay autores en la bibliografía que utilizan únicamente 3 funciones para representar los latidos [16].

La aplicación del algoritmo de búsqueda del punto de máxima simetría en el latido, especialmente cuando es aplicado a ambas derivaciones, reduce el error en la representación de los latidos (véanse Figuras 2.6 y 2.7). Estas mejoras son más pronunciadas en la segunda derivación y cuando se utilizan pocas funciones en la representación. Probablemente la razón es que la base de datos MIT-BIH [96] fue anotada sobre la primera derivación y por tanto la segunda derivación se beneficia más de esta corrección en la posición de las anotaciones. La mejora no es tan marcada al usar más funciones ya que si aumentamos el número de funciones, al darle más flexibilidad a la representación, se puede representar adecuadamente

el complejo QRS aun cuando el centro de la representación no sea el punto de máxima simetría.

Los resultados también muestran una reducción del error cuando la señal es filtrada. Por ejemplo, se puede obtener un error NRMSD del 1 % en la primera derivación sin filtrar con 18 funciones, mientras que al filtrar se obtiene este error con tan solo 12 funciones. Sobre la segunda derivación sin filtrar no se consigue alcanzar el 1 % de error NRMSD ni siquiera con 30 funciones, mientras que con filtrado es posible alcanzarlo utilizando 18 funciones. Se realizaron pruebas separando los dos tipos de filtrado utilizados. En ellas se observó que la eliminación de la deriva de línea base por separado no resultaba en una reducción significativa del error; mientras que la eliminación de solo el ruido de alta frecuencia sí producía una mejora notable. Este resultado sugiere que la representación de Hermite es más sensible al ruido de alta frecuencia que a la deriva de línea base.

Entre los artículos revisados solamente [77] presenta resultados del error en la representación de Hermite con los que es posible la comparación con nuestros resultados. En dicho trabajo se indica que los errores (según la ecuación 2.9) utilizando 3, 4, 5 y 6 funciones de Hermite son 9.7 %, 6.8 %, 5.5 % y 4.5 %, respectivamente. Estos resultados son ligeramente inferiores a los aquí obtenidos con la señal sin filtrar. Sin embargo, cuando se utiliza la señal filtrada, el error obtenido en nuestro caso es inferior. Debemos tener en cuenta para interpretar este resultado que Lagerholm et al. no indican haber usado ningún filtrado de alta frecuencia.

La gran mayoría de los autores que utilizan funciones de Hermite para representar los complejos QRS utilizan un número de funciones no mayor de 15 [57]. Esto quiere decir que aproximadamente están cometiendo un 2 % de NRMSD, lo cual pudiera parecer un error aceptable. Sin embargo, según los resultados obtenidos de BIC y AIC_c, dichos autores están representando de forma óptima un pequeño porcentaje de los latidos, en torno al 10 % (véase Figura 2.9). Y aquellos autores que utilizan 8 o menos funciones probablemente no estén representando ningún latido de forma óptima [16][77]. La razón por la que dichos autores han estado utilizando un número subóptimo de funciones para representar la morfología del complejo QRS es probablemente que en su mayoría han basado la decisión en una simple inspección visual de la representación de algunos latidos y no en criterios más objetivos. Si bien de este modo es posible obtener resultados razonables, probablemente porque el error en la representación incluso con un número pequeño de funciones es relativamente bajo (véanse

Figuras 2.6, 2.7 y 2.8), es posible que las técnicas de estos autores se beneficiasen de emplear una representación con un mayor número de funciones para los latidos.

Las dificultades asociadas a trabajar con una alta dimensionalidad, como la propuesta por BIC y AIC, hacen que la mayoría de técnicas de aprendizaje automático se comporten de forma irregular, con rendimiento en general pobre. Se debe tener en cuenta además que la base de datos utilizada consta únicamente de dos derivaciones, pero el estándar en la rutina clínica es el ECG de 12 derivaciones, lo que supondría una dimensionalidad mayor. Por ello, es habitual en estos casos recurrir a técnicas de selección de características. Sin embargo, los intentos realizados en este sentido han sido infructuosos, constatando la dificultad de aplicarlas a un problema con una alta variabilidad como el que nos concierne (véase Sección 2.1.6).

En la Figura 2.10 se aprecia cómo el tiempo requerido para el cálculo aumenta no linealmente al incrementar el número de funciones utilizadas en la representación. Mientras que para dos funciones se requieren 0.033 segundos, para 30 funciones son necesarios más de 5 segundos por latido. La implementación realizada no podría procesar ECG en tiempo real en un ordenador si el número de funciones utilizado es superior a 16. Si el procesado tuviera que ser realizado en un sistema con poca potencia de cálculo, como un microcontrolador o un teléfono, necesitaríamos de igual forma elegir un orden de funciones bajo para evitar que el tiempo de ejecución se incrementara excesivamente.

Una alternativa para reducir el error sin incrementar el coste en recursos sería utilizar un número más bajo de funciones de Hermite en la representación y corregir la posición de las anotaciones, que permite reducir el error sin incrementar significativamente la carga computacional. Por ejemplo, representar los latidos usando las anotaciones de la base de datos y sin filtrar con $N = 3, 4, 7$ conlleva unos errores de 0.112, 0.076 y 0.038 (véase Figura 2.8) con unos tiempos de ejecución por latido medios de 0.064 s, 0.103 s y 0.269 s, respectivamente. Si corregimos las anotaciones, con unos tiempos de ejecución casi idénticos, obtenemos errores que caen hasta 0.090, 0.065 y 0.031, respectivamente.

A la hora de valorar los resultados de tiempo de ejecución presentados en este capítulo debe tenerse en cuenta que la implementación realizada no fue optimizada, primando la claridad del código y la corrección por encima de la eficiencia. Por ello sería posible reducir los tiempos de ejecución optimizando la implementación. No obstante, pese a esta posible mejora, el cálculo de la representación de Hermite tal y como fue descrito en este capítulo es costoso, especialmente si se utiliza un número alto de funciones. Este hecho podría dificultar

la aplicación de esta representación al electrocardiograma. Además, es necesario considerar que la obtención de la representación es solo el primer paso en el análisis del ECG, por lo que habría que añadir el coste computacional del resto del procesado. Por ello se consideró conveniente estudiar vías alternativas para reducir el tiempo requerido para calcular la representación, concretamente mediante su paralelización empleando GPUs. Este será el objeto de estudio en el próximo capítulo.

USC
UNIVERSIDADE
DE SANTIAGO
DE COMPOSTELA

Capítulo 3

Cálculo de la representación de Hermite utilizando GPUs

En el capítulo anterior se estudió la representación del latido mediante las funciones de Hermite y se analizó el tiempo de ejecución del cálculo de dicha representación. Este tiempo puede llegar a suponer un problema de rendimiento, especialmente si se utiliza el número de funciones sugerido por AIC_c y BIC (véase Sección 2.3).

El paralelismo es una opción habitual en la bibliografía para resolver problemas científicos complejos que requieren procesar grandes cantidades de datos. Entre las muchas aproximaciones disponibles destaca el uso de GPUs (*"Graphics Processing Units"*), siendo una de las opciones más atractivas y más usadas para la aceleración de procesos computacionales. Las GPUs son relativamente baratas, fáciles de instalar y, en muchos casos proporcionan tanta potencia de cómputo como cientos de procesadores CPU trabajando en paralelo, a la vez que presentan un consumo de energía significativamente menor. Esto facilita su integración en equipos de monitorización electrocardiográfica, que mediante una GPU pueden llegar a realizar, en un tiempo equivalente, los mismos cálculos que un clúster de computación formado por docenas, o incluso cientos, de ordenadores. Su uso presenta ventajas considerables tanto en el ámbito de la monitorización hospitalaria, como en la monitorización domiciliaria, donde los datos podrían enviarse al hospital para su procesamiento. Es fácil encontrar ejemplos de aplicaciones de la tecnología de GPUs en el campo de la biomedicina: reconstrucción MRI [74], simulación tejido cardiaco [49], dinámica molecular [151], bioinformática [102], entre otras muchas.

Por ello hemos considerado interesante el estudiar cómo acelerar el cálculo de la representación de Hermite mediante el uso de GPUs. Los dispositivos GPU están diseñados para realizar una misma tarea de forma repetitiva sobre una gran cantidad de datos. Si la tarea requiere ejecuciones con flujos condicionales o muestra una gran dependencia entre los datos no se da una situación propicia para el uso de GPUs y el rendimiento total puede no mejorar notablemente, sino incluso disminuir. Previamente, en el capítulo anterior, ya se aplicó una paralelización basada en procesamiento paralelo mediante hilos para calcular la representación de Hermite. Los buenos resultados obtenidos y la idoneidad del problema para su paralelización motivaron que se decidiera utilizar GPUs para optimizar la obtención de la representación de Hermite. En la operación del cálculo de la representación de Hermite cada complejo QRS en cada derivación representa una operación independiente, que no requiere memoria compartida y que además hay que repetir una gran cantidad de veces. Es más, con un manejo hábil de las operaciones resulta una tarea que es posible paralelizar incluso dentro del mismo complejo QRS, a nivel de muestra. Por todo ello, este problema es un escenario perfecto para sacar todo el partido a las GPUs y conseguir una aceleración considerable.

3.1. Programación utilizando GPUs

Habitualmente, para programar las GPUs se utilizan lenguajes basados en C. Dichos lenguajes permiten diseñar e implementar los mecanismos de paralelización de forma fácil, permiten una compilación rápida y proporcionan una integración sencilla con otros sistemas. En este trabajo se decidió utilizar un lenguaje basado en C llamado CUDA (*"Compute Unified Device Architecture"*)[74]. El motivo principal de dicha decisión fue la facilidad de implementación y su extendido uso. El lenguaje CUDA realiza una encapsulación de los detalles del hardware, simplificando la tarea de desarrollo y permitiendo una mayor portabilidad entre distintas plataformas hardware.

Un dispositivo GPU está formado por varios multiprocesadores llamados *"streaming processors"* (SP), cada uno a su vez compuesto por varios núcleos que trabajan en paralelo. La unidad básica de ejecución en la GPU se denomina *"kernel"*. La GPU ejecuta el mismo *kernel* en paralelo con conjuntos de datos distintos como entrada. Cada hilo de ejecución ejecuta una instancia concreta del *kernel* en paralelo con el resto de hilos. Los hilos se agrupan en *"warps"* y son gestionados por un SP. Todos los hilos de un *warp* comparten el

mismo código, siguen el mismo camino de ejecución y se espera que se detengan en el mismo punto. El SP se encarga de agrupar a todos los hilos en el mismo punto de ejecución y de que estén siempre sincronizados. Por lo tanto, la presencia de bloques condicionales puede deteriorar considerablemente el rendimiento ya que el SP debe esperar a que todos los hilos alcancen el mismo punto para seguir con la ejecución.

En CUDA existen mecanismos que permiten controlar la ejecución y paralelización de los hilos. De esta forma el desarrollador puede controlar cómo se agruparán los hilos; una buena estrategia para diseñar estos grupos es fundamental para lograr aprovechar al máximo el paralelismo del procesador. Estos grupos que forma el desarrollador se denominan *bloques* y cada uno estará representado por un identificador (ID). De forma similar los *bloques* se pueden agrupar a su vez en *grids*, esto es, conjuntos de *bloques* que también tienen su propio identificador. Finalmente, cada hilo también tendrá un identificador que, combinado con el identificador del *bloque* y del *grid*, permite indexar los hilos de forma fácil y sencilla. Combinando estos identificadores cada hilo puede generar localizaciones en memoria únicas a las que acceder de forma privada. Durante la ejecución cada *bloque* es asignado a un SP que gestiona su ejecución.

Respecto a la memoria, las GPUs tienen una jerarquía de memoria piramidal. La memoria de mayor tamaño es una memoria global (DRAM) a la que pueden acceder todos los hilos, seguida de una memoria compartida (SRAM) a la que pueden acceder todos los hilos de un mismo *bloque* y finalmente una memoria privada, que está formada por los registros privados de cada hilo de ejecución. Como suele ser habitual, cuanto mayor es la memoria mayor es el tiempo de acceso necesario. La memoria global, con una gran capacidad (1-6 GB) es la más lenta, mientras que la memoria compartida es pequeña (16-48 KB) pero mucho más rápida (unos dos órdenes de magnitud más rápida que la memoria global). En la memoria global el acceso es en bloque, ya que las operaciones de lectura y escritura manejan bloques de bits consecutivos (32, 64, 128, etc.). También se dispone de una caché de nivel 2 que optimiza el uso de la memoria global, pero ello no evita que se deba realizar una planificación apropiada del uso de la memoria. Una mala planificación conllevaría un uso ineficiente de la memoria global y por consiguiente un mayor tiempo de procesamiento. El acceso a la memoria compartida se realiza de forma aleatoria, por lo que mientras se observen ciertas restricciones de uso el acceso será rápido.

Teniendo todo esto en cuenta, es posible diseñar un algoritmo para calcular una representación de Hermite de forma que aproveche al máximo las capacidades de una GPU.

Para conseguir esto es necesario realizar una cuidadosa elección de los parámetros y diseñar concienzudamente el flujo de ejecución y de datos del algoritmo.

3.2. Implementación optimizada en C

Antes de pasar a diseñar el algoritmo para calcular la representación de Hermite del QRS utilizando CUDA, se realizará una implementación optimizada en C. Con esta implementación ejecutada en una CPU estableceremos una implementación de referencia con la que comparar la aceleración de la implementación en CUDA (CPU vs GPU). Esto nos permite establecer una referencia más justa para la comparación que la implementación previa en Java, que no estaba optimizada.

Los cálculos necesarios para realizar la caracterización de los complejos QRS mediante Hermite y el pseudocódigo optimizado se muestran en el Algoritmo 1. Como entrada el algoritmo recibe el registro de ECG y el número de funciones de Hermite a utilizar en la representación. Las salidas son el conjunto de características que representarán al latido: los coeficientes de las funciones de Hermite que mejor representan el latido y la σ utilizada.

El Algoritmo 1 consta de tres partes principales: i) extracción de los complejos QRS, ii) precomputación de las funciones de Hermite (*#Lazo1*) y, iii) cálculo de los coeficientes de las funciones y de σ (*#Lazo2*). Para la extracción de los complejos QRS se usa el mismo procedimiento ya detallado en la Sección 2.1.3, tomando 144 muestras para representar cada complejo QRS (suponiendo frecuencia de muestreo de 360Hz).

En el bucle (*#Lazo1*, líneas 4-6) se calculan los valores de las funciones de Hermite $\phi_n[l,\sigma)$ para todos los σ posibles mediante la Ecuación (2.3). Estas funciones serán utilizadas intensivamente en el siguiente bucle, de ahí el interés en calcularlas previamente y almacenar el resultado, evitando así repetir el cálculo. Para medir el incremento en rendimiento de este cálculo se midió la diferencia entre el tiempo necesario para ejecutar el precomputado de las funciones junto con el segundo bucle (*#Lazo2*) y el tiempo necesario para ejecutar el segundo bucle (*#Lazo2*) calculando las funciones dentro del bucle. La alternativa de precalcular los valores consiguió una aceleración de 105x comparada con la otra opción, ejecutando las pruebas en un procesador Intel-i7. Este test demuestra que resulta beneficioso realizar este cálculo previamente.

Por otra parte, en el bucle (*#Lazo2*, líneas 8-21) se realiza el cálculo para obtener el conjunto de características (los coeficientes $c_n(\sigma)$ y la σ) que mejor representan el latido. En

Algoritmo 1 Caracterización del complejo QRS sin paralelizar

[tbp]

Input: Registro de ECG, N (número funciones a utilizar)
Output: σ y $c_n(\sigma)$ que minimizan el error para cada latido

1: **for all** latido i **do**
2: Extraer señal del complejo QRS, $x_i[l]$, del ECG.
3: **end for**
4: **for all** n y σ **do** #Lazo1
5: Calcular $\phi_n[l,\sigma)$ #Ecuación (2.3)
6: **end for**
7: $\text{Err}_{min} = \infty$
8: **for all** latido i, $x_i[l]$ **do** #Lazo2
9: **for all** σ **do**
10: **for all** n **do**
11: Calcular $c_n(\sigma)$ #Ecuación (2.8)
12: **end for**
13: Calcular $\hat{x}_i[l] = \sum_{n=0}^{N-1} c_n(\sigma)\phi_n[l,\sigma)$ #Ecuación (2.2)
14: Calcular MSE Err=$\sum_l (e[l])^2 = \sum_l (x_i[l]\text{-}\hat{x}_i[l])^2$ #Ecuación (2.7)
15: **if** Err<Err_{min} **then**
16: $\text{Sigma}_{best} = \sigma$
17: $\text{C}_{best} = c_n(\sigma)$
18: Err_{min} =Err
19: **end if**
20: **end for**
21: **end for**

esta parte tenemos tres bucles anidados. El más externo (línea 8) itera sobre todos los latidos extraídos. El segundo (línea 9) prueba iterativamente todos los posibles valores de σ y el más interno calcula los coeficientes de las N funciones de Hermite utilizadas (véase Ecuación (2.8)). Tras este cálculo, el segundo bucle selecciona los valores σ y $c_n(\sigma)$ óptimos (aquellos que minimizan el MSE (2.7)).

Analizando el Algoritmo 1 puede verse que tanto (*#Lazo1*) como (*#Lazo2*) permiten una paralelización casi completa. En la siguiente sección intentaremos paralelizar ambos bucles adaptando este algoritmo al uso de una GPU de la forma más optimizada posible.

Algoritmo 2 Caracterización del complejo QRS paralelizada con la GPU

[tbp]

Input: Registro de ECG, N (número funciones a utilizar)
Output: σ y $c_n(\sigma)$ que minimizan el error para cada latido

1: **for all** latido i **do**
2: Extraer señal del complejo QRS, $x_i[l]$, del ECG.
3: **end for**
4: Reservar memoria en la GPU
5: $kernel\_\phi(N)$
6: Enviar los complejos $x_i[l]$ a la GPU #Escribir en la memoria de la GPU
7: Esperar a que la GPU termine de procesar
8: $kernel\_Hermite(x_i[l],N)$ #Algoritmo 3
9: Esperar a que la GPU termine de procesar
10: Leer σ y $c_n(\sigma)$ para cada latido #Escribir en la memoria del ordenador el resultado

3.3. Implementación paralela

Partiendo del código del Algoritmo 1, se realizó una paralelización de (*#Lazo1*) y (*#Lazo2*) por medio de dos *kernels*: *kernel_*ϕ, encargado de precalcular los valores de las funciones de Hermite, y *kernel_Hermite*, encargado de calcular el valor de los coeficientes y el valor del error. El nuevo flujo de ejecución se muestra en el Algoritmo 2. En este caso el algoritmo fue diseñado para aprovechar al máximo las capacidades de la GPU.

El primer paso en la versión paralela es extraer los complejos QRS y reservar la memoria necesaria para almacenarlos en la GPU (Algoritmo 2, líneas 1-4). Tras esto, se llama al primer *kernel* (*kernel_*ϕ) para generar los valores de las funciones de Hermite (Algoritmo 2, línea 5) y los resultados son almacenados en la GPU. Dicho *kernel* será ejecutado en la GPU y al mismo tiempo, mientras está siendo ejecutado, los latidos son enviados a la memoria global de la GPU (Algoritmo 2, línea 6). Posteriormente, cuando la GPU termine de procesar el primer *kernel* se pasará al segundo (*kernel_Hermite*) (Algoritmo 2, línea 8). Tras terminar esta ejecución los resultados finales son enviados a la memoria del ordenador (Algoritmo 2, líneas 9-10).

Este será el esquema general de procesado del algoritmo en la GPU. En las siguientes secciones se concretará en detalle la función de cada uno de los dos *kernels*, además de presentar la estrategia empleada para optimizar la transferencia de datos para el procesado de latidos en tiempo real.

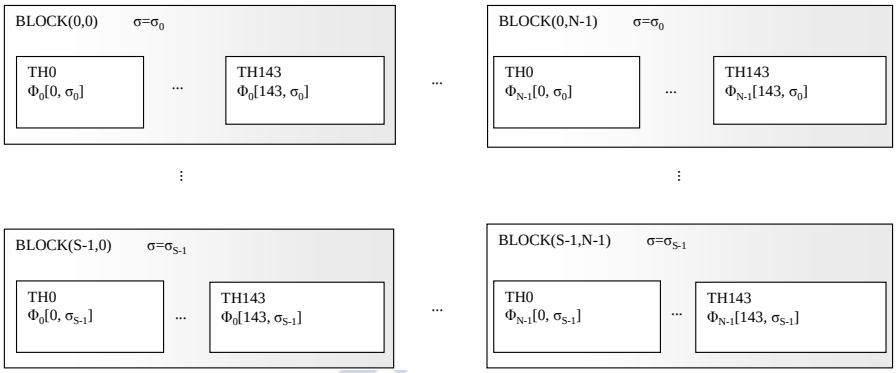


Figura 3.1: Distribución de hilos en *bloques* para el *kernel_*ϕ.

3.3.1. Precomputación de las funciones de Hermite

El *kernel_*ϕ, equivalente al (*#Lazo1*) del Algoritmo 1, es paralelizable directamente sin necesidad de realizar cambios en el código. En el caso de trabajar a 360 Hz, como en la base de datos MIT-BIH Arrhythmia Database, la configuración óptima utiliza *bloques* de 144 hilos, correspondientes a las 144 muestras que representan cada complejo QRS. Los *bloques* se distribuirán en un *grid* de dimensiones $S \times N$, siendo S el número de posibles valores que toma σ y N el número de funciones de Hermite a calcular.

Cada *bloque* se encarga del cálculo de todas las muestras para una función de Hermite $\phi_n[l, \sigma)$ con un valor concreto de σ y n. El *bloque* tendrá un identificador bidimensional, siendo su primera componente un índice referido al valor de σ dentro de S que debe calcular y su segunda componente un índice referido a la función de Hermite. Cada hilo dentro de un *bloque* se encarga de evaluar la Ecuación (2.3) para una muestra distinta (correspondiente al identificador del hilo). De esta forma se consigue una implementación completamente paralela, calculando simultáneamente el valor de tantas funciones como SP haya en la GPU (véase Figura 3.1).

3.3.2. Caracterización del complejo QRS mediante Hermite

En el Algoritmo 1 el núcleo de la caracterización del latido está en el (*#Lazo2*) donde se obtienen las características que representan al complejo QRS. La paralelización de este

proceso en el *kernel_Hermite* requerirá de un análisis más pormenorizado que el realizado para el *kernel_*ϕ.

En este caso cada *bloque* trabajará sobre un complejo QRS concreto y tendremos por tanto tantos *bloques* como latidos en el registro (aproximadamente unos 2000 latidos por registro de la base de datos MIT-BIH Arrhythmia Database). Posteriormente se abordará cómo procesar registros de larga duración como los Holter. Cada *bloque* deberá contar con tantos hilos como muestras en el complejo QRS (144 por *bloque* con una frecuencia de muestreo de 360 Hz). Con esta estructura sabremos en todo momento sobre qué latido se están realizando los cálculos (ID del *bloque*) y sobre qué muestra (ID del hilo).

El Algoritmo 3 muestra el pseudocódigo desarrollado para ejecutar el *kernel_Hermite*. Este mismo código se ejecuta para todos los hilos, por ello lo primero será identificar sobre qué latido y qué muestra se va a trabajar (Algoritmo 3, líneas 1-2). Seguidamente la información del complejo QRS será copiada a la memoria compartida, ya que va a ser utilizada múltiples veces. Esto implica que toda referencia posterior a $x_i[l]$ será realizada mediante una lectura rápida de la memoria compartida. De forma análoga al Algoritmo 1, se utiliza un bucle para recorrer todos los posibles valores de σ (Algoritmo 3, líneas 5-23). Para cada valor se calcula el vector de coeficientes $c_n(\sigma)$ (Algoritmo 3, líneas 6-11). Se debe tener en cuenta que la multiplicación (Algoritmo 3, línea 7) se realiza de forma totalmente paralela y que el resultado se guarda en una variable compartida entre los hilos. Posteriormente los resultados parciales obtenidos por cada uno de los hilos se suman para obtener el coeficiente de la función de Hermite; esta suma (Algoritmo 3, línea 10) se realiza mediante la técnica de suma por reducción [74], para evitar realizarla de forma completamente secuencial. Posteriormente, se calcula la representación del latido obtenida a partir de los coeficientes $c_n(\sigma)$ (Algoritmo 3, líneas 12-15). Con esta representación se calculará el error MSE (Algoritmo 3, líneas 16-17). De nuevo la multiplicación es paralela y la suma se realiza mediante la técnica de reducción. Finalmente, solo el hilo 0 actualiza la mejor solución si el MSE calculado es menor que el mínimo actual (Algoritmo 3, líneas 18-22).

3.3.3. Optimización de la transferencia de datos

El flujo de datos presentado en el Algoritmo 2 puede ser optimizado si se tiene en cuenta la capacidad de la GPU de realizar al mismo tiempo un cálculo y una transferencia de datos (de la memoria del ordenador a la GPU o de la GPU al ordenador). Esto resulta especialmente interesante al lidiar con el análisis de registros de ECG muy grandes (por ejemplo, registros

Algoritmo 3 *kernel_Hermite*

[tbp]

Input: Muestras de ECG de un latido $x_i[l]$, N (número funciones a utilizar)
Output: σ y $c_n(\sigma)$ que minimizan el error para el latido

```
i = bloque.ID  # ID del bloque usado como índice de latido
l = hilo.ID  # ID del hilo usado como índice de muestra
Copiar x_i[l] a la memoria compartida  # Copia completamente paralela
Err_min = ∞
for all σ do
    for all n do
        Calcular varTempHilo_l = x_i[l] φ_n[l, σ]  #Ecuación (2.8)
    end for
    for all n do
        Calcular c_n(σ) = (Σ_l varTempHilo_l)  #Suma paralela por reducción
    end for
    x̂_i[l] = 0
    for all n do
        Calcular x̂_i[l] += c_n(σ) φ_n[l, σ]  #Ecuación (2.2)
    end for
    Calcular e[l] += (x_i[l]-x̂_i[l])^2  #Ecuación (2.7)
    Calcular MSE = Σ_l e[l]  # Suma paralela por reducción
    if l== 0 y MSE <Err_min then  # Solo entrará en el condicional el hilo 0
        Sigma_best = σ
        C_best = c_n(σ)
        Err_min =MSE
    end if
end for
```

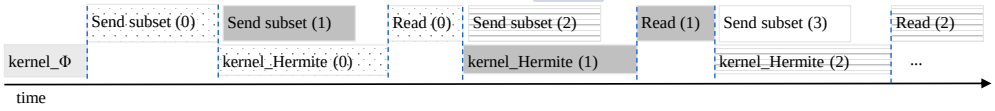


Figura 3.2: Esquema del flujo de trabajo optimizado de la GPU.

Holter). Dichos registros contienen una gran cantidad de latidos que no caben en la memoria de la GPU y que hacen necesario un procesamiento por lotes en conjuntos de latidos más pequeños.

Por otra parte, en el caso de procesamiento en tiempo real también es necesario que los latidos sean procesados en pequeños conjuntos para evitar una gran latencia en la presentación

de los resultados. La Figura 3.2 muestra la forma de optimizar el tiempo de ejecución mediante el solapamiento de las transferencias de datos con los cálculos realizados por la GPU. La primera tarea en la GPU siempre será la ejecución del *kernel_*ϕ. Posteriormente se procesarán secuencialmente pequeños conjuntos de latidos en el *kernel_Hermite*. El primer conjunto de latidos (subset 0) será enviado tras la ejecución del *kernel_*ϕ. A continuación, mientras está siendo procesado el *kernel_Hermite*, se enviará a la GPU el siguiente conjunto de latidos (subset 1). Al terminar la ejecución del primer conjunto (subset 0) se leerá el resultado de la ejecución y se transferirá a la memoria del ordenador. Seguidamente se pasará a ejecutar en *kernel_Hermite* el segundo conjunto (subset 1) mientras se lee el próximo (subset 2). De esta forma se aprovecha al máximo la GPU y cada vez que se está realizando una operación sobre un conjunto en *kernel_Hermite* se está leyendo el próximo conjunto de datos a procesar.

El flujo de ejecución presentado permite un procesamiento constante en tiempo real, estando la GPU siempre calculando la representación de un conjunto de latidos y optimizando el rendimiento. Se debe hacer notar al lector que las duraciones de las tareas mostradas en la Figura 3.2 no son reales, primando en este caso la claridad de la representación sobre la exactitud. La duración de la ejecución del *kernel_Hermite* siempre sobrepasará el tiempo dedicado a la transferencia de datos, pero la proporcionalidad entre las dos tareas no es la mostrada en la gráfica.

3.4. Resultados y Discusión

En esta sección se presentarán los resultados obtenidos al aplicar la paralelización por medio de GPUs en tres escenarios distintos y se compararán con la implementación de referencia en C. La implementación de referencia fue codificada siguiendo el Algoritmo 1, mientras que la implementación en GPU sigue el esquema planteado en el Algoritmo 2, incluyendo la paralelización del cálculo realizada por *kernel_Hermite* (Algoritmo 3) y la optimización de transferencia de datos detallada en la Sección 3.3.3. Las ejecuciones de todas las pruebas fueron realizadas utilizando un ordenador Intel Core i7 (1,6GHz y 4GB de RAM) con una GPU TESLA C2050 (448 núcleos, 4GB de RAM). Todos los resultados se obtuvieron tomando latidos de la base de datos MIT-BIH Arrhythmia Database (véase Sección 1.3.1).

Los tres escenarios que se consideraron en las pruebas fueron los siguientes:

Test A **Procesado en diferido (offline) de registros cortos.** En este escenario se pretende estudiar el comportamiento del algoritmo cuando se procesa una pequeña cantidad de datos. Para ello se tomaron conjuntos de latidos de distinto tamaño M. En concreto se realizaron pruebas con cuatro tamaños: $M \in \{10, 10^2, 10^3, M_{max}(N)\}$, donde $M_{max}(N)$ es el máximo número de latidos que puede procesar la GPU al mismo tiempo para un número de funciones de Hermite N (aproximadamente 5000). Se realizaron estas pruebas para distintos valores de N, $N \in \{6, 10, 20, 30\}$. En total se llevaron a cabo 16 experimentos con las distintas combinaciones de M y N. Si bien este escenario no se corresponde con una situación realista en la rutina clínica, proporciona una perspectiva útil para comprender el comportamiento de la GPU.

Test B **Procesado en diferido (offline) de registros largos.** En este caso se pretendía estudiar el rendimiento cuando se procesan registros de ECG de larga duración, como los registros Holter. Se realizaron pruebas con un número de latidos M de aproximadamente 10^5 (unas 16 horas). Los latidos tuvieron que ser procesados por la GPU en subconjuntos de alrededor de 5000 latidos, siempre intentado procesar el máximo número de latidos permitidos por la GPU para cada valor de N. Se realizaron para este escenario 4 experimentos en total con distintos valores de N, $N \in \{6, 10, 20, 30\}$.

Test C **Procesado en tiempo real (online).** Este escenario pretende simular un procesamiento en tiempo real del ECG, como el que se realiza habitualmente en una Unidad de Cuidados Intensivos. Los latidos se procesaron en subconjuntos de pequeña duración (1, 5, 10 y 100 latidos). El número de subconjuntos de latidos a procesar fue elegido de tal forma que el número total de latidos siempre fuera lo más cercano posible a 10^5. Se realizaron 16 experimentos distintos combinando los distintos tamaños del bloque de procesamiento con los valores asignados a N, $N \in \{6, 10, 20, 30\}$.

3.4.1. Test A: Procesado en diferido (offline) de registros cortos

En la Tabla 3.1 se muestra el tiempo de computación y la aceleración conseguida (comparada con la implementación de referencia) en este escenario para los distintos experimentos realizados. La primera columna de la tabla indica el número de funciones utilizadas para la representación (N) y la segunda columna el número de latidos procesados (M). El número de latidos máximos procesados cambia para cada N ya que depende del

Tabla 3.1: Resultados para el Test A del tiempo de ejecución para cada uno de los experimentos y aceleración obtenida.

N	M	CPU (ms)	GPU (ms)	Aceleración
6	10	26	66	0.39x
	100	173	65	2.65x
	1000	1637	79	20.66x
	5000	10626	128	83.02x
10	10	59	67	0.87x
	100	282	71	3.95x
	1000	2569	82	31.32x
	4800	15534	155	100.15x
20	10	177	75	2.38x
	100	604	80	7.52x
	1000	4868	122	39.86x
	4600	27287	288	94.90x
30	10	372	75	4.93x
	100	995	89	11.24x
	1000	7216	180	39.99x
	4400	37454	521.12	71.87x

número máximo que puede procesar al mismo tiempo la GPU. La tercera y cuarta columnas muestran respectivamente el tiempo de computación en ms de la implementación de referencia (CPU) y de implementación paralela (GPU). Finalmente, la última columna muestra la aceleración conseguida con la GPU (tiempo en la CPU dividido tiempo en la GPU). Una aceleración menor que la unidad implica que hemos empeorado el rendimiento con la implementación paralela.

En la Figura 3.3 se muestra el tiempo de ejecución necesario para ambas implementaciones frente al número de latidos para los distintos valores de N. Esta gráfica ilustra de manera más clara la dependencia que existe entre el tiempo de computación con el número de funciones utilizadas y el número de latidos procesados.

Los resultados obtenidos muestran claramente cómo la aceleración proporcionada por la GPU mejora al incrementar el número de funciones utilizadas, el número de latidos procesados o ambos a la vez. Para registros cortos la GPU no proporciona un beneficio considerable, e incluso en ocasiones tiene un rendimiento peor que el de la CPU (aceleración menor que 1). Las mejoras más notables se obtienen al procesar registros con un número de

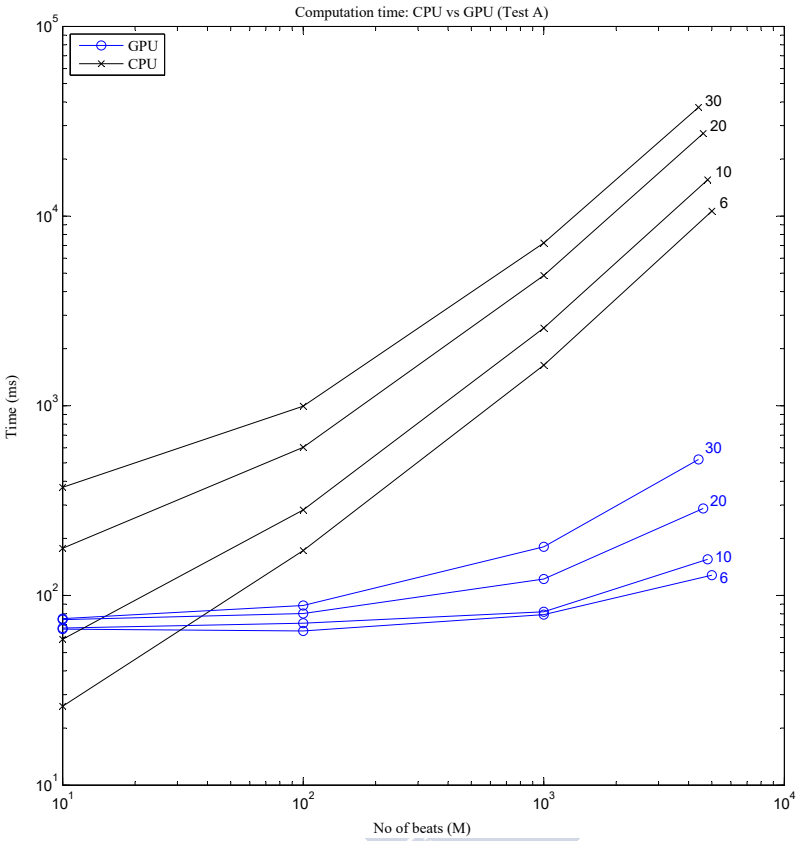


Figura 3.3: Tiempo de ejecución para ambas implementaciones en el Test A.

latidos elevado (mayor que 1000), obteniéndose aceleraciones entre 72x y 100x con los conjuntos de latidos más grandes.

Se realizó un análisis pormenorizado de la distribución del tiempo de ejecución entre las distintas tareas en la implementación paralela (Algoritmo 2). En la Figuras 3.4 y 3.5 se muestra dicha distribución para $N = 6$ y $N = 60$, respectivamente, con porcentajes del tiempo total dedicado a:

1. Reserva de memoria en el PC (Malloc CPU)
2. Reserva de memoria en la GPU (Malloc GPU)
3. Ejecución del *kernel_*ϕ
4. Transferencia de latidos a la GPU (1st Transfer)
5. Ejecución del *kernel_Hermite*
6. Transferencia de los resultados al ordenador (Read)

La distribución de tiempos entre estas tareas cuando se utilizan 6 funciones de Hermite y con distintos valores de M se muestra en la Figura 3.4. Como se puede apreciar, el mayor porcentaje de tiempo corresponde a la tarea de reserva de memoria en la GPU. Para un número de latidos pequeño es mucho mayor el porcentaje de tiempo dedicado a preparar la GPU para el procesamiento que el porcentaje de tiempo dedicado a procesar los latidos. Cuando se incrementa el número de latidos, incrementa el porcentaje de tiempo asociado al cálculo de los coeficientes de Hermite (*kernel_Hermite*). El coste asociado al cálculo de los valores de las funciones de Hermite (*kernel_*ϕ) es casi constante, debido a que al no cambiar N el número de operaciones a realizar serán las mismas; por tanto, cuanto más latidos haya y más grande sea el tiempo de procesado total el porcentaje del tiempo de ejecución de esta tarea será menor. Dado que la implementación de referencia tiene una inicialización mucho más rápida, no compensa utilizar la GPU para un conjunto pequeño de latidos. No obstante, en la GPU al incrementar el número de latidos el porcentaje de tiempo dedicado al procesado supera al porcentaje necesario para la inicialización. Dado que la CPU requiere mucho más tiempo para realizar el procesado de la representación de Hermite, es posible obtener una aceleración significativa al utilizar la implementación paralela con GPU en vez de la implementación de referencia en C cuando trabajamos con conjuntos grandes de datos.

En la Figura 3.5 se muestran resultados similares, pero en este caso utilizando 30 funciones de Hermite. El porcentaje de tiempo de computación frente al de inicialización es mayor y por tanto redunda en una mayor aceleración (véase Tabla 3.1). Nótese también que

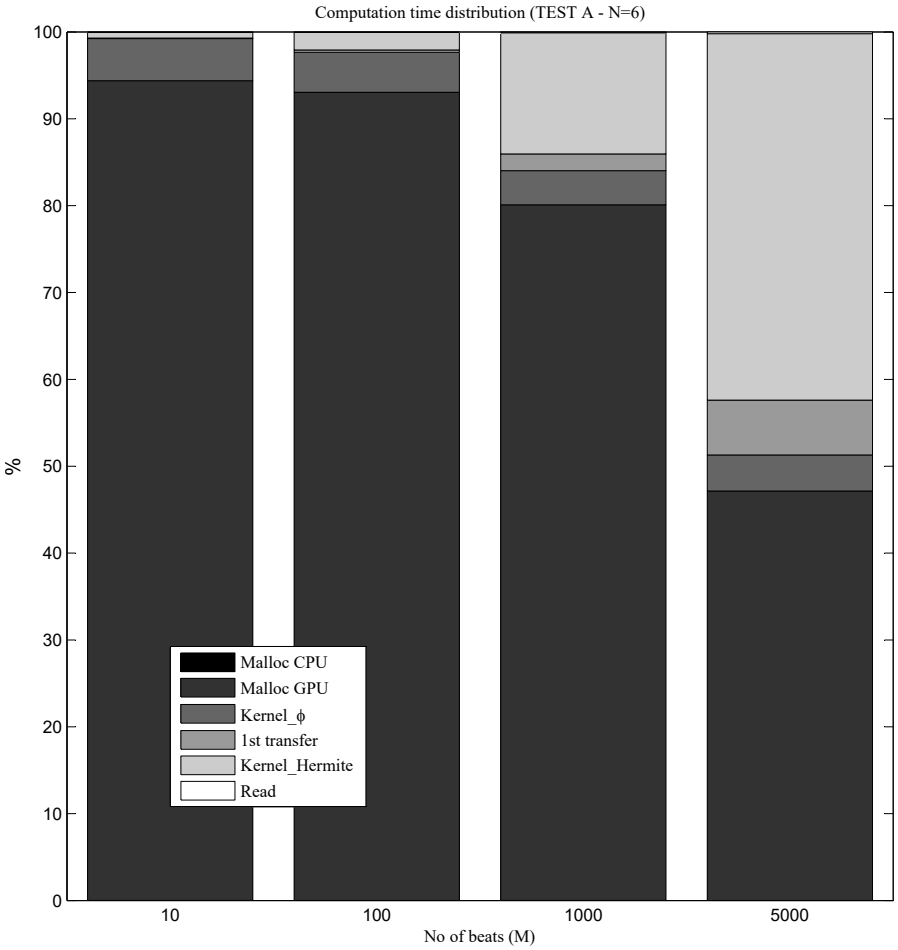


Figura 3.4: Porcentajes de tiempo de ejecución dedicados a cada tarea en Test A con $N = 6$.

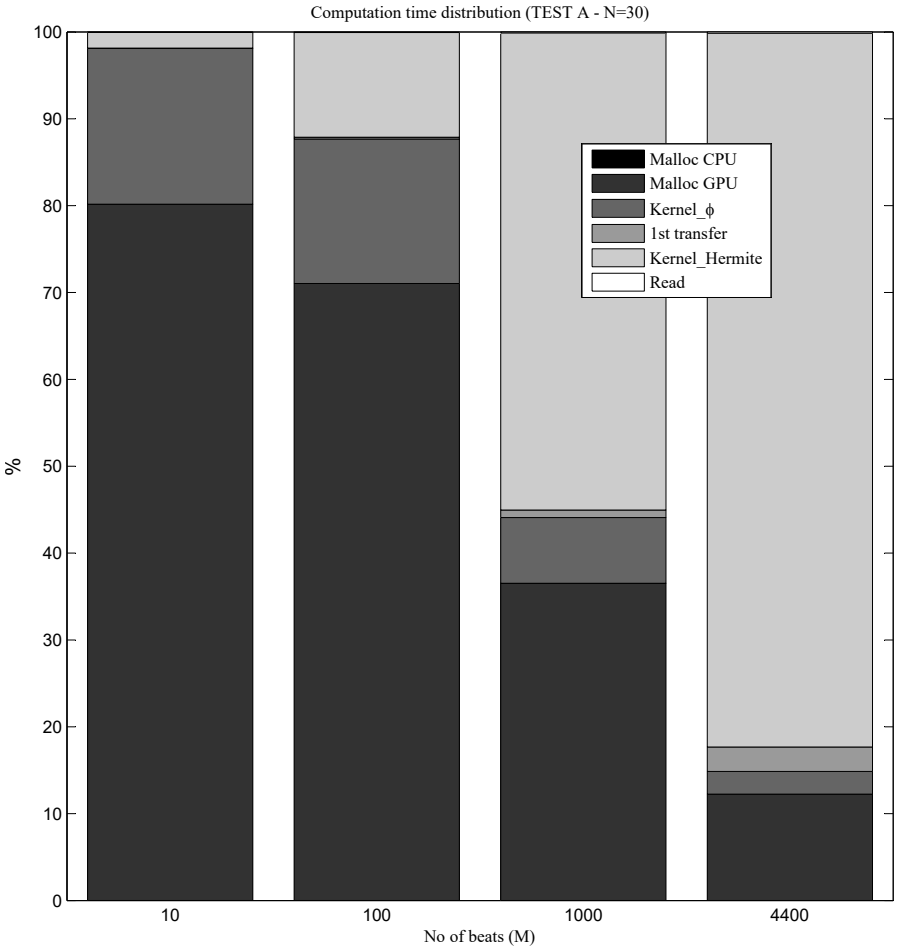


Figura 3.5: Porcentajes de tiempo de ejecución dedicados a cada tarea en Test A con $N = 30$.

Tabla 3.2: Resultados para el Test B del tiempo de ejecución para cada uno de los experimentos y aceleración obtenida.

N	M1	M2	CPU (ms)	GPU (ms)	Aceleración
6	20	5000	211559	1230	172x
10	21	4762	322829	1735	186x
20	22	4546	599249	4612	130x
30	23	4348	845513	9882	86x

el tiempo de transferencia para registros cortos tampoco es desdeñable en este caso. Por otra parte, el tiempo necesario para calcular las funciones de Hermite aumenta al haber aumentado N, pero permanece constante independientemente del número de latidos y por tanto pierde importancia en registros largos.

Debe resaltarse que el rendimiento no se incrementa siempre de forma monótona (véase Tabla 3.1). La causa de esto es que para valores de $N > 20$ el número de variables locales en el *kernel* excede el número de registros en el SP y por tanto es necesario recurrir a la memoria global, incurriendo en el llamado *register spilling* [101]. Esto supone un impedimento importante para conseguir un rendimiento óptimo en la GPU.

3.4.2. Test B: Procesado en diferido (offline) de registros largos

El tiempo de procesado necesario y la aceleración lograda en este escenario se muestran en la Tabla 3.2. En la primera columna de la tabla se indica el número de funciones utilizadas para la representación (N). En la segunda y tercera columnas se indican el número de bloques de latidos procesados ($M1$) y el número de latidos por bloque ($M2$). El número total de latidos (M) se puede obtener multiplicando ambos datos ($M = M1 \cdot M2$) y es siempre de aproximadamente 10^5 latidos. La cuarta y quinta columnas contienen, respectivamente, el tiempo de la implementación de referencia (CPU) y la implementación paralelizada (GPU). Para finalizar la última columna muestra la aceleración conseguida (tiempo en la CPU dividido tiempo en la GPU).

La aceleración de la GPU es sensiblemente mayor que en el primer test. Esto concuerda con lo apreciado en los resultados del Test A, en los que el mayor rendimiento se lograba al procesar registros largos. Se puede observar cómo de nuevo para valores de N mayores la aceleración es menor, debido al ya mencionado efecto del *register spilling*. En la Figura 3.6 se muestra la distribución del tiempo de ejecución entre las distintas tareas para distintos

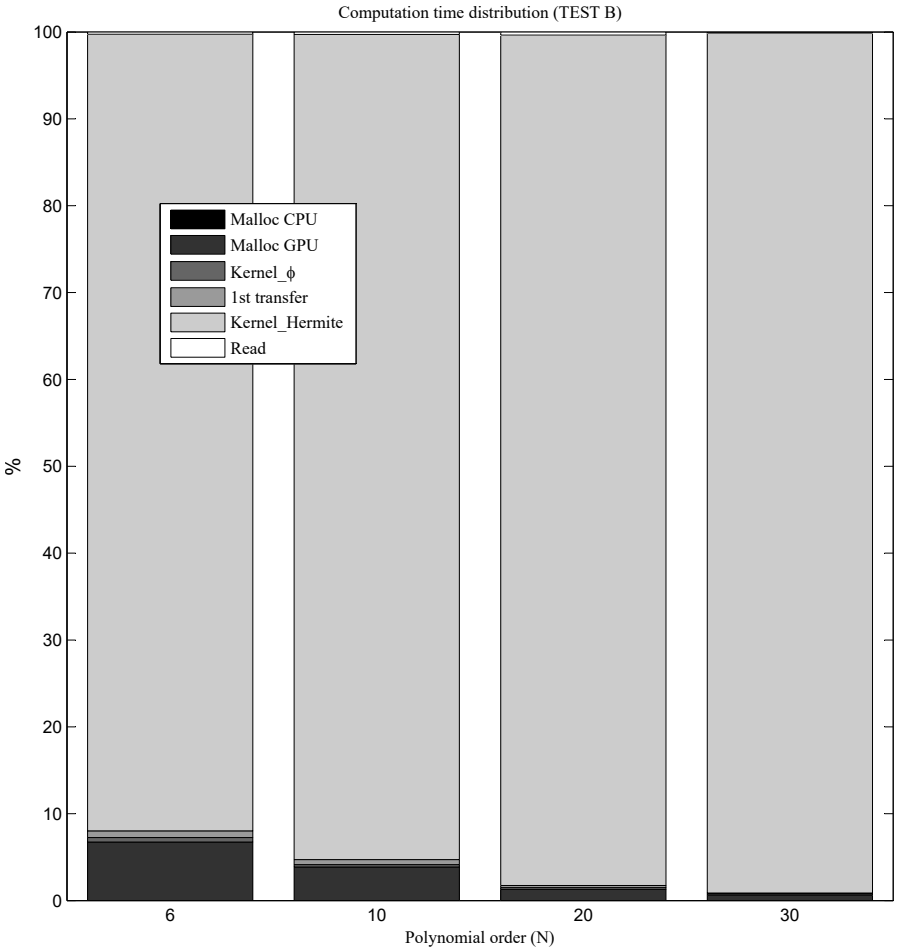


Figura 3.6: Porcentajes de tiempo de ejecución dedicados a cada tarea en el Test B para distintos valores de N.

Tabla 3.3: Resultados para el Test C del tiempo de ejecución para cada uno de los experimentos y aceleración obtenida.

N	M2	CPU (ms)	GPU (ms)	Aceleración
6	1	160652	46882	3.43x
	5		9315	17.25x
	10		5404	29.73x
	100		1494	107.53x
10	1	250691	61864	4.05x
	5		12815	19.56x
	10		6613	37.90x
	100		2267	110.56x
20	1	472305	101674	4.65x
	5		20626	22.90x
	10		10402	45.40x
	100		4965	95.12x
30	1	689779	157276	4.39x
	5		29978	23.01x
	10		15653	44.07x
	100		11359	60.72x

valores de N. Como se puede apreciar, la tarea asociada al *kernel_Hermite* ocupa ahora más del 90 % del tiempo de ejecución. En este caso no aparece representado el tiempo necesario para transferir los datos a la GPU ya que este proceso se solapa con la computación del *kernel_Hermite*, siguiendo la estrategia explicada en la Sección 3.3.3 (véase Figura 3.2).

La aceleración obtenida varía entre 86x y 172x, demostrando claramente los beneficios de una GPU para el procesado de registros de larga duración. El tiempo de procesado se reduce de minutos (CPU) a segundos (GPU). Para un registro Holter con 12 canales de 3 días, para obtener la representación de Hermite utilizando 30 funciones, pasamos de necesitar más de 3 horas y media con una CPU a necesitar dos minutos y medio. Esto permite liberar recursos computacionales que podrían ser utilizados para aplicar técnicas de análisis sobre la representación obtenida de los latidos.

3.4.3. Test C: Procesado en tiempo real (online)

Los resultados correspondientes a este último escenario se muestran en la Tabla 3.3. En la primera columna se muestra el número de funciones de Hermite utilizadas (N). En la segunda el número de latidos procesados en cada bloque ($M2$). Cuanto más alto sea $M2$, mayor será

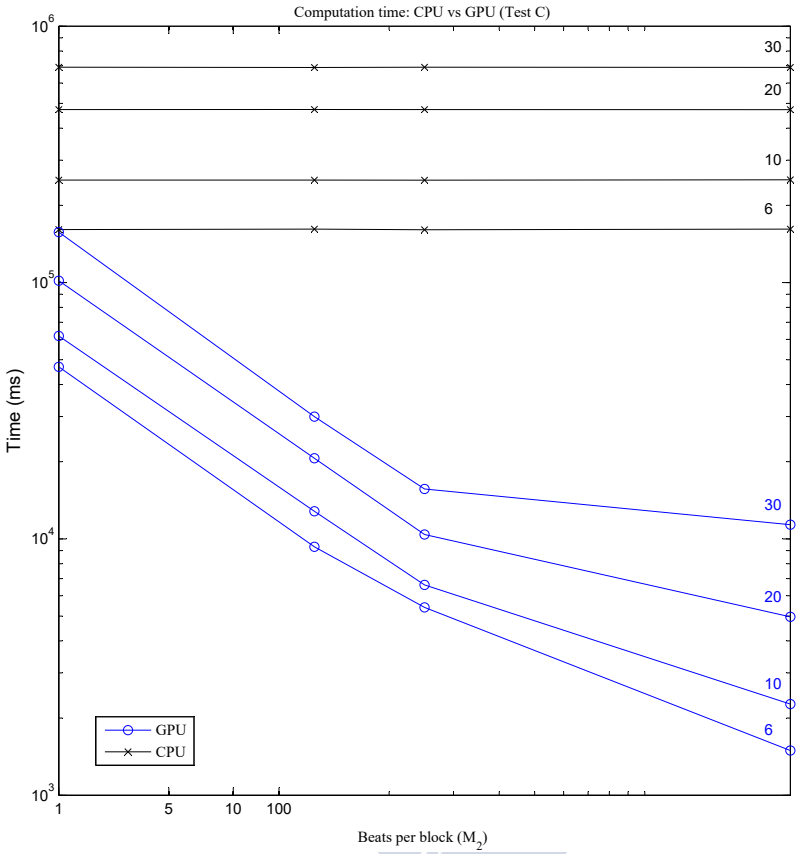


Figura 3.7: Tiempo de ejecución para ambas implementaciones en el Test C.

la latencia del sistema. Asumiendo una frecuencia cardiaca ficticia de 1 Hz (un latido por segundo), el tiempo de latencia sería igual a $M2$ segundos. El valor de $M1$ no aparece por claridad y fue establecido de tal forma que $M = M1 \cdot M2 \approx 10^5$. La tercera y la cuarta columnas muestran, respectivamente, el tiempo necesario para la ejecución de la implementación de referencia (CPU) y la implementación paralelizada (GPU). El tiempo de computación en la CPU es constante para un valor dado de N ya que no depende de $M2$ y el número de latidos M es siempre el mismo. En la última columna se muestra la aceleración obtenida (tiempo en la CPU dividido tiempo en la GPU).

La aceleración varía entre 4x y 110x, siendo sensiblemente menor cuando se impone un valor de $M2$ pequeño. Los resultados obtenidos demuestran que el tiempo de ejecución es menor al usar una GPU incluso cuando se impone una latencia mínima ($M2 = 1$). No obstante, la aceleración obtenida en este caso no parece suficiente como para justificar el uso de una GPU. Una CPU con una implementación multihilo más optimizada podría llegar a alcanzar un tiempo similar al de la GPU. Si se aumenta el tamaño de $M2$ es posible ver cómo el beneficio de utilizar una GPU aumenta rápidamente (véase Figura 3.7). Otro aspecto a tener en cuenta es que, aunque la CPU sea capaz de procesar en tiempo real la caracterización de Hermite (el tiempo de computación es menor que el tiempo que tarda el corazón en generar los latidos), se limitaría en gran medida la capacidad de computo disponible para el procesado posterior sobre esta representación. Sin embargo, si se delega el cálculo de la representación de Hermite en la GPU, es posible usar toda la capacidad de cómputo de la CPU para realizar un análisis adicional del latido.

En la Figura 3.8 se puede ver cómo se reparte el tiempo de ejecución en la implementación paralela utilizando 6 funciones de Hermite. Nuevamente, la mayor parte del tiempo lo ocupa la tarea de calcular los coeficientes de Hermite (*kernel_Hermite*). En este caso, a diferencia de los escenarios anteriores, la tarea de transferir los resultados a la memoria del ordenador ocupa un porcentaje de tiempo visible en el gráfico. Dicho porcentaje es mayor para valores pequeños de $M2$, lo que implica que para conseguir una latencia menor la GPU va a dedicar un porcentaje considerable de tiempo a esta tarea. El reparto porcentual de tiempo para los otros valores de N es similar al mostrado.

Como se puede apreciar en los resultados, hay un claro compromiso entre la latencia y la aceleración conseguida utilizando una GPU. Esto concuerda con los resultados observados en los escenarios previos, en los que la GPU mostraba mejor rendimiento cuando trabajaba con un gran número de latidos. Por tanto, para caracterizar latidos en tiempo real el uso de una GPU puede ser una alternativa viable para el cálculo de la representación de Hermite incluso cuando se un utilizan 30 funciones, suponiendo que se puedan agrupar los latidos en grupos de 5 o más latidos, lo cual implicaría una latencia de unos 5 segundos.

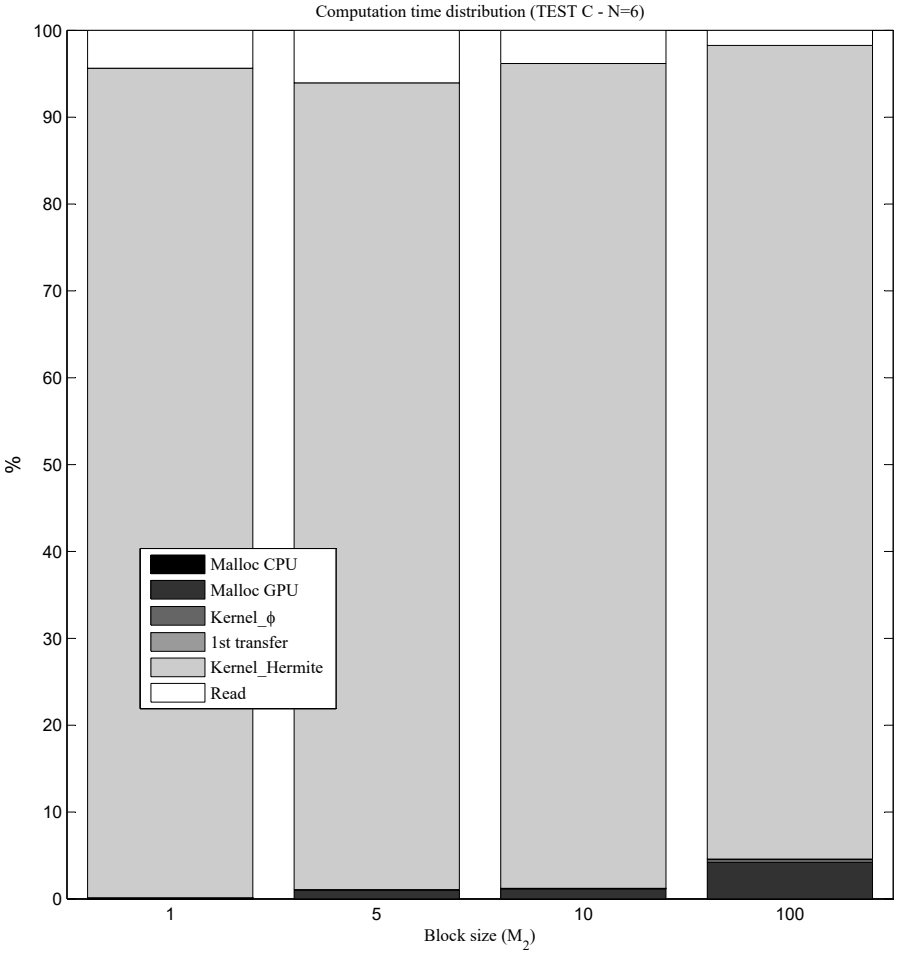


Figura 3.8: Porcentajes de tiempo de ejecución dedicados a cada tarea en el Test C para $N = 6$.

Capítulo 4

Agrupamiento de latidos mediante acumulación de evidencia

Una vez resuelto el problema de la representación del latido abordaremos el problema de su agrupamiento morfológico. Una parte importante del proceso de análisis basado en agrupamiento consiste en seleccionar la técnica apropiada para el conjunto de datos que estamos explorando. Para conseguirlo, generalmente se realizan varias pruebas con diferentes algoritmos y distintas configuraciones, hasta encontrar una que proporcione un resultado satisfactorio, basándose en la información disponible del problema. Sin embargo, este es un proceso lento, típicamente guiado por heurísticas, con una fuerte dependencia del criterio del analista y proclive a error [39].

En la bibliografía de agrupamiento de latidos no existe un consenso con respecto a la superioridad de una técnica de agrupamiento frente a otra, pudiendo obtenerse resultados similares con técnicas muy diversas [20][21][26][27][75][77][93][120] (véase Sección 1.2). Esto resulta coherente con los análisis comparativos en el ámbito del agrupamiento automático [65]; ninguna técnica de agrupamiento ha mostrado hasta ahora una superioridad manifiesta sobre las demás, sino que simplemente presenta una mejor adaptación a las características de algunos tipos específicos de problemas.

Tomando inspiración de los trabajos de combinación de clasificadores y de fusión de sensores, en los últimos años se han desarrollado múltiples técnicas de agrupamiento mediante la combinación de particiones obtenidas por distintos algoritmos de agrupamiento [11][36][42][61][138]. Estas nuevas técnicas de agrupamiento por medio de combinación,

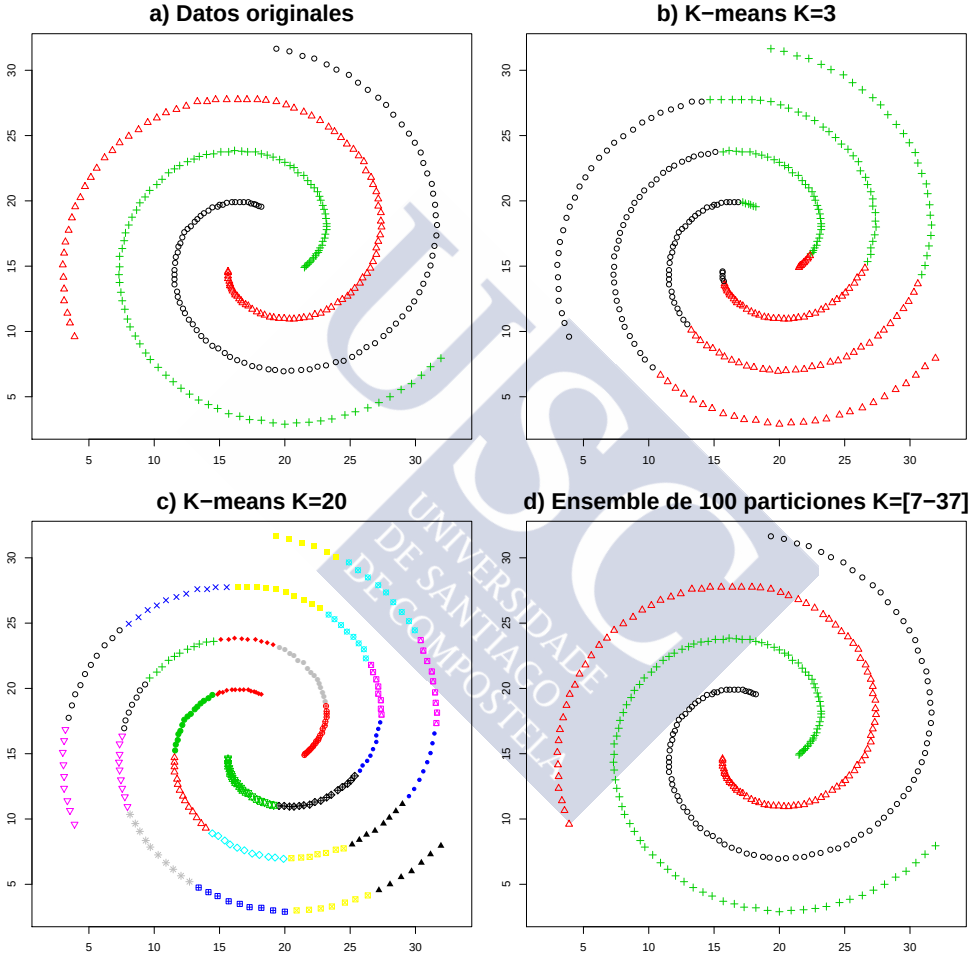


Figura 4.1: (a) muestra las tres particiones naturales de los datos; (b) y (c) muestran dos versiones del algoritmo K-means con K=3 y K=20, respectivamente; (d) muestra el resultado de la combinación de 100 particiones de datos distintas creadas por el algoritmo K-means con valores de K aleatorios entre 7 y 37.

también llamadas agrupamiento mediante *ensembles*, se han afianzado como alternativas a las técnicas de agrupamiento tradicionales, ya que a menudo son capaces de mejorar la robustez y la estabilidad de los resultados del agrupamiento. Mediante ellas es posible combinar las particiones generadas por distintos algoritmos conservando en gran parte sus ventajas individuales y contrarrestando sus desventajas. Podemos ver un ejemplo en la Figura 4.1 donde se muestra cómo combinando los resultados de 100 ejecuciones del algoritmo K-means es posible representar las tres particiones originales de los datos, particiones que nunca podrían haber sido obtenidas por una sola aplicación de K-means, dado que este algoritmo siempre obtiene particiones hiperesféricas. Otra ventaja adicional del agrupamiento mediante *ensembles* es que permite reducir la dependencia entre el algoritmo de agrupamiento y los resultados, ya que posibilita la utilización de múltiples algoritmos con diferentes parámetros de inicialización, combinando posteriormente los resultados obtenidos por cada uno de ellos.

El paradigma de acumulación de evidencia (*Evidence Acumulation Clustering*, EAC) proporciona una de las técnicas que permite realizar agrupamiento mediante *ensembles*. Propuesto por primera vez en [42], es el resultado de una búsqueda de nuevas técnicas que no impongan una forma o modelo en los grupos. Para ello combina varios resultados de agrupamiento, extrayendo la información individual de cada partición y combinándola para definir los grupos. Esta misma idea ya ha sido extensamente explorada en clasificación, resultando en mejoras en la exactitud y precisión por medio de la combinación de múltiples clasificadores [58][80][125]. En EAC se combinan los resultados individuales de varios agrupamientos para obtener una nueva medida de similitud entre las instancias, que integra y resume la información obtenida por cada uno de los algoritmos utilizados para obtener las particiones. Posteriormente, se obtendrá la partición final de los datos basándose en esta nueva medida de similitud.

En la siguiente sección se expondrá una nueva técnica de agrupamiento basada en acumulación de evidencia. Posteriormente, se aplicará dicha técnica al agrupamiento del latido según su origen cardiaco, utilizando para ello la base de datos MIT-BIH Arrhythmia Database. Finalmente se extenderá el método para usarlo con 12 derivaciones y se validará sobre la base de datos INCARTDB.

4.1. Agrupamiento mediante acumulación de evidencia positiva y negativa (PN-EAC)

En esta sección se presenta una nueva técnica de agrupamiento basada en el paradigma de acumulación de evidencia: agrupamiento mediante acumulación de evidencia positiva y negativa (*Positive and Negative Evidence Accumulation Clustering*, PN-EAC). Dicha técnica introduce un concepto novedoso que denominaremos *evidencia negativa*. En el paradigma de agrupamiento mediante acumulación de evidencia se emplea un *ensemble*, es decir, un conjunto de particiones de los datos, para extraer información de cada partición. Seguidamente se registra qué instancias aparecen en un mismo grupo en cada resultado individual, tratando esta información como evidencia de que dichas instancias deberían aparecer en el mismo grupo en la partición final. A esta información, habitualmente referida en la literatura como "evidencia", la denominaremos *evidencia positiva*. La idea subyacente es que el juicio de un comité formado por un conjunto de individuos diferentes, va a resultar más cercano a la realidad que el juicio de un único individuo. En PN-EAC además de emplear evidencia positiva, se extrae también evidencia sobre instancias que según las particiones generadas no deberían aparecer en el mismo grupo en la partición final del agrupamiento. A esta información la denominaremos *evidencia negativa*.

Formulación del problema

Sea $X = \{x_1, x_2, \ldots, x_n\}$ un conjunto de n elementos. Un algoritmo de agrupamiento tomará X como entrada y generará una *partición* de datos $P = \{A_1, A_2, \ldots, A_k\}$ de los n elementos de X divididos en k grupos, donde A_j representa el grupo j de la partición de datos. Definimos un *ensemble* como un conjunto de m particiones distintas de datos $E = \{P_1, P_2, \ldots, P_m\}$, obtenidas utilizando distintos algoritmos, distintos parámetros, o diferentes representaciones de los datos. No se realiza ninguna asunción sobre el número de grupos de cada una de las particiones del *ensemble*, pudiendo cada una tener un número de grupos k distinto. Las particiones del ensemble serán combinadas en una única partición final, P_*, que será el resultado del agrupamiento por acumulación de evidencia y que, por lo general, será más cercana a la partición natural subyacente en los datos que la mayor parte de las particiones individuales por separado. La técnica proporciona más expresividad al resultado final que cada una de las particiones, al permitir superar sus limitaciones en la geometría de la similitud.

Algoritmo 4 Esquema general del agrupamiento mediante ensembles

Input: Conjunto de elementos $X = \{x_1, x_2, \ldots, x_n\}$
Input: Parámetros: m número de particiones a generar
Output: P_* partición final de los n elementos

```
for i=1 to m do
    Generar partición P_i a partir de los n elementos de X  #Sección (4.1.1)
end for
for i=1 to m do
    Extraer y acumular evidencia de P_i  #Sección (4.1.2)
end for
Extraer la partición final de los datos P_*  #Sección (4.1.3)
```

En las siguientes secciones se detallará el funcionamiento del algoritmo PN-EAC (véase Algoritmo 4). Para ello se explicarán cómo se crean las particiones $P_1, P_2, \ldots, P_m$, cómo se combinan dichas particiones y cómo se extrae la partición final P_* de dicha combinación.

4.1.1. Generación de las particiones del *ensemble*

La utilización de técnicas de agrupamiento mediante *ensembles* requiere que los resultados de agrupamiento individuales, $P_1, P_2, \ldots, P_m$, sean distintos. El uso de diferentes representaciones de los datos, la utilización de técnicas para alterar los datos como muestreo o *"bagging"* [35], o el uso de diferentes algoritmos de agrupamiento o de diferentes parametrizaciones de dichos algoritmos producirán, en general, distintas particiones de los datos. Mediante la combinación de estas particiones la técnica de *ensembles* puede superar los resultados individuales, abstrayendo las distintas representaciones, modelos u inicializaciones para extraer la información común a todas ellas y obtener la partición final.

Entre la multitud de métodos de agrupamiento disponibles, el algoritmo K-means es uno de los más usados y destaca por su simplicidad. Su sencillez es una de sus grandes ventajas, que lo hace computacionalmente eficiente y rápido. Su reducido número de parámetros (típicamente solo el número de grupos) es otra de sus fortalezas. Por contra, su mayor limitación es la rigidez que impone en los grupos que forma, que siempre tienen forma hiperesférica. Esta limitación imposibilita la identificación de grupos con formas arbitrarias, lo que a menudo hace que sea relegado en favor de otros algoritmos más complejos. Sin embargo, como ya se demostró en [43], mediante la combinación de varias ejecuciones de K-means podemos superar esta limitación (véase Figura 4.1).

En este trabajo se hará uso de K-means para generar las particiones de datos debido principalmente a su eficiencia y bajo coste computacional. La generación de las distintas particiones se realizará mediante la ejecución del algoritmo con inicializaciones aleatorias de los centroides iniciales y diferentes valores para el número de grupos a crear. Este número de grupos k será obtenido aleatoriamente para cada ejecución en el rango dado por:

$$k \in [\sqrt{n}/2, \sqrt{n}] \tag{4.1}$$

siendo n el número de instancias en el conjunto de datos. Dado que K-means es un algoritmo voraz, incluso empleando el mismo valor para el parámetro k pueden obtenerse resultados diferentes si la inicialización de los centroides es distinta.

4.1.2. Combinación de las particiones de datos

En la literatura de agrupamiento por medio de *ensembles* es posible encontrar varias aproximaciones para combinar los resultados de distintas particiones de datos [51][137][138]. En este caso se adoptó la propuesta de [42] basada en la acumulación de evidencia. En dicha aproximación se utiliza un mecanismo de voto para recoger la información de aquellas parejas de instancias que aparecen en un mismo grupo. El mecanismo de voto no hace ninguna asunción acerca de los resultados individuales, por lo que sería posible combinar particiones con distinto número de grupos o incluso incompletas.

Evidencia Positiva

La información de voto será recogida en una matriz G, de tamaño $n \times n$. Para cada partición individual, la co-ocurrencia de un par de instancias i y j en el mismo grupo será almacenada como un voto positivo en la celda correspondiente de la matriz G. La idea subyacente es que las instancias que pertenecen a un mismo grupo "natural" es más probable que sean asignadas al mismo grupo en las distintas particiones $P_1, P_2, \ldots, P_m$.

La matriz G, consideradas las m particiones, tomará la siguiente forma:

$$G_{(i,j)} = \frac{n_{ij}}{m}, \tag{4.2}$$

dónde n_{ij} es el número de veces que la pareja de instancias i y j ha sido asignada al mismo grupo en las particiones $P_1, P_2, \ldots, P_m$.

Evidencia Negativa

En algunas ocasiones, debido a la representación de los datos, el hecho de que dos instancias aparezcan en el mismo grupo de una partición puede no aportar demasiada información acerca del agrupamiento natural de los datos. Esta evidencia, que se podría considerar "débil", podría incluso introducir ruido en la matriz de evidencia, empeorando el resultado final. Por ejemplo, supongamos que queremos agrupar personas en base a sus hábitos de consumo y que cada persona está representada por su edad y su salario anual. Con esta representación, el hecho de que dos personas pertenezcan a un mismo grupo proporciona una información relativamente pobre acerca de sus hábitos de consumo: es perfectamente posible que dos personas de edad similar y con un salario similar tengan hábitos de consumo bastante diferentes (por ejemplo, en base a si están o no casados, o si tienen o no hijos). Sin embargo, el hecho de que dos personas pertenezcan a distintos grupos, lo que implica que o bien tienen una marcada diferencia en edad, en salario, o en ambos, sí proporciona evidencia de que esas dos personas probablemente difieran en sus hábitos de consumo: típicamente los hábitos de consumo se ven bastante influenciados por la edad y poder adquisitivo de la persona.

A menudo en el agrupamiento basado en *ensembles* esta situación se resuelve ignorando estas características en la representación de los datos. Sin embargo, en estos escenarios el hecho de que dos instancias no estén en el mismo grupo de una partición podría aportar información valiosa acerca de que esas dos instancias no deberían agruparse juntas en el resultado final. De esta forma, aunque estas características aporten una evidencia "débil" para agrupar instancias, pueden aportar información valiosa para saber qué instancias no debemos agrupar juntas. A esta evidencia la llamaremos *evidencia negativa*. Las particiones de las que se extrae esta evidencia las denominaremos $P_1^-, P_2^-, \ldots, P_l^-$ para distinguirlas de las particiones de las que se extrae evidencia positiva.

Para recoger esta evidencia negativa también se hará uso de un sistema de votación y una matriz, con la diferencia de que en este caso se computará un voto negativo en la celda correspondiente por cada partición en la que las instancias i y j no estén en el mismo grupo. La matriz de recolección de evidencia negativa G^-, de tamaño $n \times n$, tomará la siguiente forma:

$$G^-_{(i,j)} = -\frac{o_{ij}}{l}, \tag{4.3}$$

dónde o_{ij} es el número de veces que las instancias i y j son asignadas a grupos distintos en las particiones $P_1^-, P_2^-, \ldots, P_l^-$.

Es importante resaltar que las particiones de las que se extrae evidencia negativa no deben ser las mismas utilizadas para obtener evidencia positiva. De cada partición se puede extraer un tipo de información, pero no ambas. De hecho, las particiones utilizadas para obtener evidencia negativa deberían ser generadas sobre representaciones de los datos diferentes que las particiones empleadas para generar evidencia positiva, representaciones que deberían ser adecuadas para este tipo de evidencia. También es necesario mencionar que la evidencia negativa solo puede ser utilizada en conjunto con la evidencia positiva. La evidencia negativa aporta información sobre qué instancias no deberían ser agrupadas juntas, funcionando como una serie de restricciones, pero esta información solo es útil conjuntamente con información de qué instancias sí deben agruparse juntas, información proveniente de la evidencia positiva.

Tras extraer la evidencia positiva y, si procede, la evidencia negativa, el algoritmo PN-EAC combina ambas matrices en una única matriz de evidencia, G^*, que será utilizada para generar la partición final de los datos:

$$G^* = G + G^- . \tag{4.4}$$

4.1.3. Extracción de la partición final de los datos

El último paso del agrupamiento por medio de *ensembles* consiste en extraer la partición final de los datos P_* de la matriz de evidencia G^*, que en este caso se comporta como una matriz de similitud entre las instancias. Para obtener dicha partición final se aplicará un algoritmo jerárquico de enlace sobre la matriz de evidencia [44]. Dicho algoritmo está diseñado para aplicarse sobre matrices de similitud y, por tanto, es adecuado para trabajar sobre la matriz de evidencia G^* [129].

El agrupamiento jerárquico es un método que busca establecer una jerarquía de grupos, conformándolos como si de un árbol se tratase [28]. Para ello se pueden seguir dos estrategias distintas: aglomerativa, en la que cada instancia comienza siendo un grupo y en cada iteración se juntan dos grupos; o divisiva, en la que todas las instancias comienzan en el mismo grupo y en cada iteración hay un grupo que se divide. Los resultados generalmente se presentan en forma de dendrograma, esto es, como un árbol que muestra los distintos grupos y cómo se unen y dividen. El criterio de división o unión suele estar basado en una métrica entre las instancias de cada grupo; en nuestro caso se utilizará la distancia euclídea. También tendremos que decidir cómo calcular la distancia entre dos grupos, si calculando la media de todas las

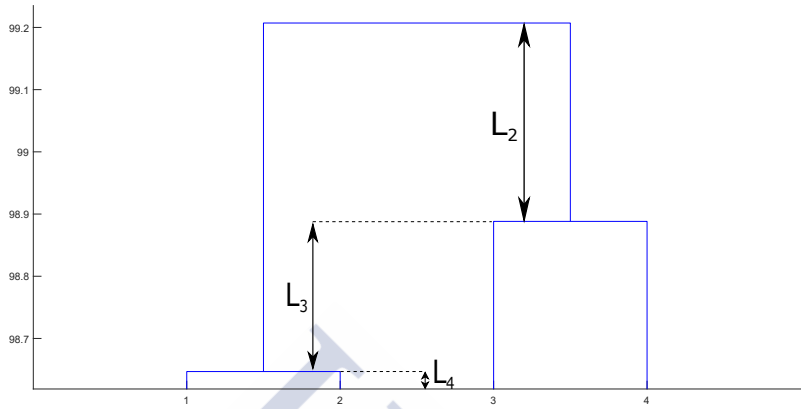


Figura 4.2: Ejemplo de dendrograma para ilustrar el criterio del tiempo de vida y los valores para 2 ($L_2 = 0.31$), 3 ($L_3 = 0.23$) y 4 ($L_4 = 0.02$) grupos.

distancias entre sus instancias (enlace medio), o tomando la mayor (enlace completo), o la menor (enlace simple) o alguna otra de entre las alternativas existentes [99].

La mayor parte de los algoritmos basados en *ensembles* dejan la elección del número final de grupos en manos del usuario, quien decide "cortar" el dendrograma a una determinada altura para generar un número determinado de grupos. En [43] se propone una alternativa: el uso de un criterio basado en el tiempo de vida de cada grupo en el dendrograma. En un algoritmo jerárquico aglomerativo, como el aquí propuesto, cada instancia comienza como un grupo independiente; posteriormente en cada iteración los dos grupos más cercanos son fusionados, hasta quedar un solo grupo. Podemos definir el tiempo de vida de una cantidad de grupos k como la diferencia absoluta entre los valores del dendrograma en los que ocurre la transición de $k+1$ a k grupos y la transición de k a $k-1$ grupos (véase Figura 4.2). Se repetirá esta operación para todos los posibles valores de k (los posibles números de grupos) y se escogerá aquella k con un mayor valor para este criterio. En la Figura 4.2 mostramos un ejemplo: aparecen representados los valores del tiempo de vida para 2, 3 y 4 grupos, L_2, L_3 y L_4, respectivamente; en este caso concreto el mayor valor es L_2 y por tanto se elige $k = 2$. Esta idea se basa en que aquella partición más similar a la partición natural de los datos será la más estable respecto a la fusión o división de grupos en el dendrograma; es decir, tendrá el mayor tiempo de vida.

4.1.4. Análisis de complejidad computacional de PN-EAC

Para estudiar el rendimiento de PN-EAC se realizó un análisis de la complejidad de cada una de sus partes. La complejidad del algoritmo K-means utilizado para crear las particiones (véase Algoritmo 4, línea 2) ha sido ampliamente estudiada en la bibliografía [52]: $\mathcal{O}(n \cdot k \cdot d \cdot i)$, siendo i el número de iteraciones, d el número de dimensiones del vector de características y k el número de grupos. El número de dimensiones para la mayor parte de los problemas será fijo y mucho menor que n. Lo mismo ocurre con el número de iteraciones i, que estará limitado por un máximo fijo (típicamente 100). El número de grupos k en nuestro algoritmo se determina según (4.1) por lo que como mucho será $\sqrt{n}$. Por lo tanto, la complejidad de K-means puede aproximarse por: $\mathcal{O}(n \cdot k \cdot d \cdot i) \approx \mathcal{O}(n \cdot \sqrt{n})$. Este algoritmo será ejecutado una vez por cada partición (véase Algoritmo 4, líneas 1-3) por lo que la complejidad total de este paso del algoritmo sería $\mathcal{O}(m \cdot n \cdot \sqrt{n})$, siendo m el número de particiones creadas.

La extracción de la evidencia de las particiones requiere recorrer la matriz de evidencia, por lo que la complejidad está determinada por el tamaño de la matriz $(n \times n)$, siendo $\mathcal{O}(n^2)$. Dicha matriz se recorre una vez por cada partición generada, por lo que la complejidad computacional de recopilar la evidencia es $\mathcal{O}(m \cdot n^2)$ (véase Algoritmo 4, líneas 4-6).

Finalmente, debemos analizar la complejidad del algoritmo jerárquico utilizado para extraer la partición final. En dichos algoritmos se comienza por calcular la matriz de distancias entre todos los objetos, operación que tiene una complejidad de $\mathcal{O}(n^2)$. Posteriormente, se realizan n iteraciones y en cada una de ellas se encuentran los dos grupos más cercanos, que se unen, y se actualiza la distancia del resto de grupos con el nuevo grupo. La complejidad de cada una de estas iteraciones es de $\mathcal{O}(n \cdot log(n))$, por lo que al ser n iteraciones la complejidad total es $\mathcal{O}(n^2 \cdot log(n))$, siendo este el término que determina la complejidad del algoritmo jerárquico [28] (véase Algoritmo 4, línea 7).

La complejidad del algoritmo jerárquico ($\mathcal{O}(n^2 \cdot log(n))$) podría llegar a ser superior a la de extraer la evidencia ($\mathcal{O}(m \cdot n^2)$), en el caso de que $log(n)$ sea mayor que m. Sin embargo, para valores razonables de n y m se cumple que $m >> log(n)$. Por ejemplo, para 300 particiones n tendría que ser mayor que 2^{300}. Por ello, la complejidad de PN-EAC será $\mathcal{O}(m \cdot n^2)$:

$$\mathcal{O}(m \cdot n \cdot \sqrt{n}) + \mathcal{O}(m \cdot n^2) + \mathcal{O}(n^2 \cdot log(n)) \approx \mathcal{O}(m \cdot n^2). \tag{4.5}$$

En cuanto a la complejidad espacial, vendrá dada por el tamaño de la matriz de evidencia y será $\mathcal{O}(n^2)$, que en una implementación optimizada podría reducirse a $\mathcal{O}(n^2/2)$, al ser esta matriz simétrica.

4.2. Agrupamiento de latidos con PN-EAC

Una de las mayores dificultades en el agrupamiento de latidos reside en la complejidad morfológica del propio latido, de modo que la variabilidad intrínseca a cualquier familia morfológica, es decir, al conjunto de latidos que comparten un mismo origen de activación y camino de propagación, no se proyecta en un conjunto de características que muestren propiedades de simetría en el espacio de representación. Así es que, sea cual sea la fórmula adoptada para representar el latido, se ha comprobado que las distintas familias morfológicas muestran un conjunto de formas heterogéneas en el espacio de representación. La técnica propuesta en la sección anterior (PN-EAC) propone un agrupamiento en el que la forma de los grupos se define mediante la combinación de resultados individuales, permitiendo para cada grupo una forma diferente y arbitraria. Además, dicha técnica permite combinar de forma natural información de distintas fuentes, lo que permitiría combinar fácilmente información de distintas derivaciones.

Para el agrupamiento se utilizará la representación de Hermite del latido, ya presentada en el Capítulo 2. Esta representación solamente captura información sobre el complejo QRS. En ciertos tipos de arritmias, como contracciones prematuras auriculares y atrioventriculares, y latidos de escape auriculares y de unión, la morfología del complejo QRS es similar, y no es suficiente para discriminar y agrupar correctamente los latidos (véanse Figuras 1.6 y 4.3). En estos casos es necesaria información adicional, que idealmente se obtiene mediante la identificación de la onda P. Sin embargo, la dificultad de delinear el latido y extraer la onda P con precisión hace que sea habitual recurrir a obtener dicha información por otras vías [20][31], como en nuestro caso, que se obtendrá mediante información relativa a la distancia entre latidos. Por lo tanto, para permitir al algoritmo distinguir entre ciertos tipos de latidos cuyos complejos QRS tienen una forma muy similar se incluyeron dos características basadas en la distancia entre latidos:

$$R_1[i] = R[i] - R[i-1], \tag{4.6}$$

$$\begin{aligned} R_2[i] &= u(\alpha) \cdot \alpha, \\ \alpha &= (R_1[i+1] - R_1[i]) - (R_1[i] - R_1[i-1]), \end{aligned} \tag{4.7}$$

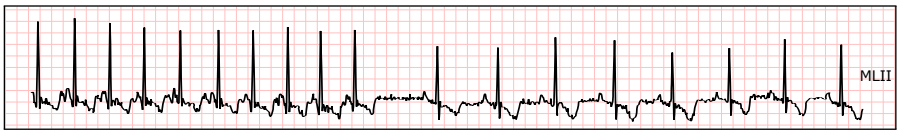


Figura 4.3: Fragmento de ECG en el que al principio aparecen latidos prematuros atrioventriculares (J) y posteriormente, aproximadamente en la mitad del fragmento, las distancias entre latidos se incrementan y aparecen latidos normales. (Fuente: MIT-BIH Arrhythmia Database Arrhythmia Database, registro 234, entre 0:14:27 y 0:14:37)

donde $R[i]$ es el momento de ocurrencia del latido número i, dado por las anotaciones de la base de datos, y $u(x)$ es la función escalón Heaviside:

$$u(x) = \begin{cases} 1 & \text{si } x \geq 0 \\ 0 & \text{si } x < 0 \end{cases} \tag{4.8}$$

Esta función permite eliminar los valores negativos poniéndolos a cero y conservar los positivos en la ecuación (4.7).

La expresión (4.6) mide la distancia de un latido con el latido previo. Comparando los valores de esta medida es posible identificar si un latido es prematuro. Para ello, la expresión (4.7) compara esta distancia con otras dos distancias: la distancia del latido actual con el siguiente y la distancia del latido previo con el anterior a este. De esta forma se normaliza esta medida para que no dependa del ritmo cardíaco. En consecuencia, un valor alto para (4.7) proporciona evidencia de que el latido puede ser prematuro. Por tanto, nuestro latido estará representado por los coeficientes de las funciones de Hermite para cada una de las derivaciones, los parámetros σ y las características dadas por (4.6) y (4.7).

Se propone un conjunto de experimentos para aplicar el algoritmo PN-EAC al agrupamiento de latidos. Para estos experimentos se limitará el número de funciones de Hermite utilizadas en la representación a 16. Se utilizará la base de datos MIT-BIH Arrhythmia Database completa (véase Sección 1.3.1), eliminando tanto el ruido de alta frecuencia como la deriva de línea base, mediante los filtros ya presentados en la Sección 2.1.1. Sin embargo, para facilitar la comparación con otros trabajos de la bibliografía, no se aplicará corrección alguna sobre las posiciones de los latidos de la base de datos, utilizando por tanto las posiciones originales.

Cada latido será representado por 36 características, 16 coeficientes de Hermite por cada derivación de la base de datos, un valor de σ por cada derivación y las dos medidas basadas en la distancia entre latidos, dadas por (4.6) y (4.7).

4.2.1. Estrategias para la generación de particiones

A la hora de aplicar el paradigma de acumulación de evidencia al agrupamiento de latidos se emplearán tres estrategias distintas para generar los *ensembles* y agrupar los latidos.

Estrategia 1 La primera estrategia es la aproximación clásica de aprendizaje automático en la que todas las características disponibles de una instancia conforman un único vector. Por tanto, en el vector aparecerán los parámetros de Hermite (coeficientes y valor de σ) para las dos derivaciones y las características dadas por las ecuaciones (4.6) y (4.7):

$$u_i = (C^1, \sigma^1, C^2, \sigma^2, R_1, R_2), \tag{4.9}$$

donde u_i es el vector de características que representa el latido x_i, C^d son los coeficientes de Hermite que representan el complejo QRS en la derivación d, $C^d = \{c_0(\sigma^d), c_1(\sigma^d), ..., c_{N-1}(\sigma^d)\}$ (en este caso con $N = 16$) y σ^d es el valor de σ en dicha derivación. Utilizando la representación indicada en la ecuación (4.9) para cada latido se generarán las particiones de datos $P_1, P_2, \ldots, P_m$ (véase Algoritmo 5, líneas 1-3).

Estrategia 2 La segunda estrategia se basa en la hipótesis de que generando particiones con distintas representaciones de datos por separado es posible obtener mejores resultados que agrupando todas las características en el mismo vector. Como ya se vio en la Sección 1.1.3, los cardiólogos suelen apoyarse para el diagnóstico en la interpretación de varias derivaciones de forma paralela. La segunda estrategia imita este *modus operandi* extrayendo evidencia por separado de la información procedente de cada derivación. También se separa la información de las características dadas por (4.6) y (4.7) al representar una información distinta a la morfológica, con otro marco de referencia. Siguiendo esta estrategia, se dividirán las características en tres representaciones diferentes del latido: una representación para la información morfológica extraída de cada derivación, representada mediante las características de Hermite, y una tercera representación con la información de las características

Algoritmo 5 Agrupamiento de latidos con PN-EAC (Estrategia 1)

Input: Conjunto de latidos $X = \{x_1, x_2, \ldots, x_n\}$
Input: m número de particiones a generar
Output: P_* partición final de los n elementos

1: **for all** latido x_i **do**
2: $u_i \leftarrow$(representación para el latido x_i según la Ecuación (4.9))
3: **end for**
4: $U = \{u_1, u_2, \ldots, u_n\}$
5: **for** j=1 **to** m **do** #Sección 4.1.1
6: $k = aleatorio([\sqrt{n}/2, \sqrt{n}])$ #Ecuación (4.1)
7: $P_j = K - means(U, k)$
8: **end for**
9: $G = zeros(n, n)$ #Inicializar matriz evidencia
10: **for** j=1 **to** m **do** #Sección 4.1.2
11: **for** h=1 **to** n **do**
12: **for** f=1 **to** n **do**
13: **if** u_h y u_f están en un mismo grupo en P_j **then**
14: $G_{(h,f)} = G_{(h,f)} + 1$ #Ecuación (4.2)
15: **end if**
16: **end for**
17: **end for**
18: **end for**
19: $G^* = G/m$ #Ecuación (4.2)
20: $P_* = enlace.medio(G^*)$ #Sección 4.1.3

derivadas de la distancia entre los latidos. Por consiguiente, tendremos tres vectores de características:

$$v_i^1 = (C^1, \sigma^1), \tag{4.10}$$

$$v_i^2 = (C^2, \sigma^2), \tag{4.11}$$

$$v_i^3 = (R_1, R_2), \tag{4.12}$$

dónde v_i^1, v_i^2 y v_i^3 son los vectores de características de cada una de las tres representaciones del latido x_i. Dichas representaciones se utilizarán por separado para generar particiones de las que extraer evidencia positiva y usando esta evidencia se extraerá la partición final (véase Algoritmo 6, líneas 1-5).

Estrategia 3 La tercera estrategia es una variante de la segunda, pero en esta ocasión se aplicará el algoritmo PN-EAC, extrayendo evidencia negativa de las características

Algoritmo 6 Agrupamiento de latidos con PN-EAC (Estrategia 2)

Input: Conjunto de latidos $X = \{x_1, x_2, \ldots, x_n\}$
Input: m número de particiones a generar
Output: P_* partición final de los n elementos

for all latido x_i **do**
  $v_i^1 \leftarrow$ (representación para el latido x_i según la Ecuación (4.10))
  $v_i^2 \leftarrow$ (representación para el latido x_i según la Ecuación (4.11))
  $v_i^3 \leftarrow$ (representación para el latido x_i según la Ecuación (4.12))
end for
$V1 = \{v_1^1, v_2^1, \ldots, v_n^1\}$
$V2 = \{v_1^2, v_2^2, \ldots, v_n^2\}$
$V3 = \{v_1^3, v_2^3, \ldots, v_n^3\}$
for j=1 **to** m **do** #Sección 4.1.1
  $P_j^1 = K - means(V1, k = aleatorio([\sqrt{n}/2, \sqrt{n}]))$
  $P_j^2 = K - means(V2, k = aleatorio([\sqrt{n}/2, \sqrt{n}]))$
  $P_j^3 = K - means(V3, k = aleatorio([\sqrt{n}/2, \sqrt{n}]))$
end for
$G = zeros(n, n)$ #Inicializar matriz evidencia
for j=1 **to** m **do** #Sección 4.1.2
  for h=1 **to** n **do**
    for f=1 **to** n **do**
      if v_h^1 y v_f^1 están en un mismo grupo en P_j^1 **then**
        $G_{(h,f)} = G_{(h,f)} + 1$ # Ecuación 4.2
      end if
      if v_h^2 y v_f^2 están en un mismo grupo en P_j^2 **then**
        $G_{(h,f)} = G_{(h,f)} + 1$ # Ecuación 4.2
      end if
      if v_h^3 y v_f^3 están en un mismo grupo en P_j^3 **then**
        $G_{(h,f)} = G_{(h,f)} + 1$ # Ecuación 4.2
      end if
    end for
  end for
end for
$G^* = G/(3 \cdot m)$ # Ecuación 4.2
$P_* = enlace.medio(G^*)$ #Sección 4.1.3

Figura 4.4: Se muestra un fragmento de ECG con cuatro latidos normales seguidos de cuatro latidos con bloqueo de rama derecha. Resaltar que la distancia entre latidos es similar para los 8, teniendo los latidos patológicos aproximadamente los mismos valores de las características (4.6) y (4.7) que los latidos normales. (Fuente: MIT-BIH Arrhythmia Database, registro 212, entre 0:12:13 y 0:12:18)

derivadas de los instantes de ocurrencia de los latidos (v_i^3). El hecho de que los valores de (4.6) y (4.7) no cambien, es decir, que no haya cambios en la distancia entre latidos, no implica que no haya un cambio en el tipo de latido; existe una gran variedad de tipos de latido que pueden darse con distancias iguales entre latidos. Por lo tanto, que los valores de estas características sean similares no proporciona información concluyente de si dos latidos pertenecen al mismo tipo (véase Figura 4.4). Sin embargo, dos latidos con diferencias considerables en estas características generalmente pertenecen a distintos tipos de latido (véase Figura 4.5). En esta estrategia para modelar esta información obtenida de las ecuaciones (4.6) y (4.7) se utilizará la evidencia negativa. Se emplearán las mismas tres representaciones del latido cardiaco que en la estrategia anterior (véanse Ecuaciones (4.10), (4.11) y (4.12)), pero en este caso de la representación derivada de la distancia entre latidos (4.12) se extraerá evidencia negativa, mientras que de las otras dos se continuará extrayendo evidencia positiva (véase Algoritmo 7, líneas 1-5).

Para la generación de las particiones se utilizará el algoritmo K-means con diferentes inicializaciones y un número de grupos aleatorio en el rango dado por (4.1) (véase Sección 4.1.1). En la primera estrategia, donde todas las características del latido están en un mismo vector u_i, se generan 300 particiones de datos. De forma similar, para la segunda y tercera estrategias se generan 100 particiones por cada vector de características (v_i^1, v_i^2 y v_i^3), con lo que tenemos en total 300 particiones por cada estrategia. En la segunda estrategia se extrae evidencia positiva de las tres representaciones dadas por las ecuaciones (4.10), (4.11) y (4.12).

Algoritmo 7 Agrupamiento de latidos con PN-EAC (Estrategia 3)

Input: Conjunto de latidos $X = \{x_1, x_2, \ldots, x_n\}$
Input: m número de particiones a generar
Output: P_* partición final de los n elementos

for all latido x_i **do**
  $v_i^1 \leftarrow$ (representación para el latido x_i según la Ecuación (4.10))
  $v_i^2 \leftarrow$ (representación para el latido x_i según la Ecuación (4.11))
  $v_i^3 \leftarrow$ (representación para el latido x_i según la Ecuación (4.12))
end for
$V1 = \{v_1^1, v_2^1, \ldots, v_n^1\}$
$V2 = \{v_1^2, v_2^2, \ldots, v_n^2\}$
$V3 = \{v_1^3, v_2^3, \ldots, v_n^3\}$
for j=1 **to** m **do** #Sección 4.1.1
  $P_j^1 = K - means(V1, k = aleatorio([\sqrt{n}/2, \sqrt{n}]))$
  $P_j^2 = K - means(V2, k = aleatorio([\sqrt{n}/2, \sqrt{n}]))$
  $P_j^3 = K - means(V3, k = aleatorio([\sqrt{n}/2, \sqrt{n}]))$
end for
$G = zeros(n, n)$ #Inicializar matriz evidencia
$G^- = zeros(n, n)$ #Inicializar matriz evidencia negativa
for j=1 **to** m **do** #Sección 4.1.2
  for h=1 **to** n **do**
    for f=1 **to** n **do**
      if v_h^1 y v_f^1 están en un mismo grupo en P_j^1 **then**
        $G_{(h,f)} = G_{(h,f)} + 1$ #Ecuación (4.2)
      end if
      if v_h^2 y v_f^2 están en un mismo grupo en P_j^2 **then**
        $G_{(h,f)} = G_{(h,f)} + 1$ #Ecuación (4.2)
      end if
      if v_h^3 y v_f^3 **no** están en un mismo grupo en P_j^3 **then**
        $G^-_{(h,f)} = G_{(h,f)} - 1$ #Ecuación (4.3)
      end if
    end for
  end for
end for
$G = G/(2 \cdot m)$ #Ecuación (4.2)
$G^- = G^-/m$ #Ecuación (4.3)
$G^* = G + G^-$ #Ecuación (4.4)
$P_* = enlace.medio(G^*)$ #Sección 4.1.3

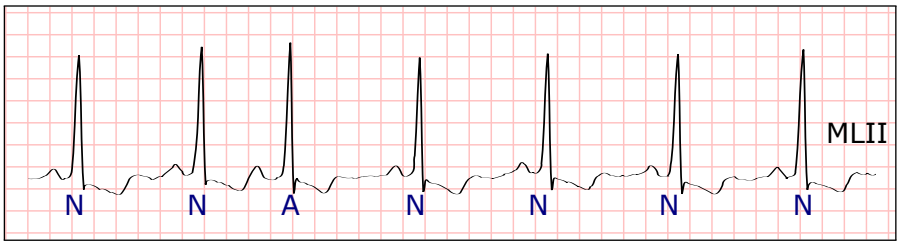


Figura 4.5: En este fragmento de ECG el tercer latido es un latido auricular prematuro, mientras que el resto son latidos normales. Se puede apreciar cómo el latido prematuro es morfológicamente similar a los normales, pero la distancia con el latido anterior y el siguiente cambia. (Fuente: MIT-BIH Arrhythmia Database, registro 223, entre 0:01:02 y 0:01:07)

En la tercera estrategia, de la representación derivada de la distancia entre latidos (4.12) se extrae únicamente evidencia negativa (Algoritmo 7, líneas 25-27). En el caso de esta última estrategia, para obtener la matriz final de evidencia se combina la evidencia positiva y la negativa mediante la ecuación (4.4) (Algoritmo 7, línea 33).

La partición final de los datos P_* se extrae de la matriz de evidencia aplicando el algoritmo de enlace medio (Algoritmo 5, línea 20; Algoritmo 6, línea 31; Algoritmo 7, línea 34). En nuestro caso, para el agrupamiento de los latidos, esta variante es la que ha mostrado el mejor rendimiento. El número de grupos de esta partición final se determina utilizando el criterio del tiempo de vida. También se obtiene el resultado con un número fijo de 25 grupos por registro.

4.2.2. Resultados

Para evaluar el rendimiento del agrupamiento se considerará que cada grupo obtenido pertenece al tipo de latido mayoritario dentro del grupo, de acuerdo con la anotación de la base de datos, considerando por tanto a todos los latidos distintos del tipo mayoritario del grupo como errores. En la rutina clínica, si fuera necesario, la correspondencia entre grupos y tipos de latidos podría obtenerse de un cardiólogo que anotara un latido de cada grupo.

En la Tabla 4.1 se muestran los resultados para las tres estrategias, registro a registro. Para cada registro aparece el número de errores con 25 grupos por registro (25C) y el número de grupos y de errores cuando se utiliza el criterio del tiempo de vida (Tiempo de vida). Al final de la tabla aparece el número total de errores y el porcentaje que representan sobre el total de la base de datos (109966 latidos). Al utilizar 25 grupos por registro se obtienen unos porcentajes de error de 2.25 %, 1.81 % y 1.44 % para la primera, segunda y tercera estrategias,

respectivamente. Al utilizar el criterio del tiempo de vida los porcentajes son de 2.75 %, 3.81 % y 0.99 % con un total de 508, 203 y 6947 grupos, respectivamente.

Tabla 4.1. Resultados del agrupamiento para las tres estrategias. Se muestra, para cada estrategia, para cada registro (#), el número de errores (Err) utilizando un número fijo de 25 grupos (25C) y utilizando el criterio del tiempo de vida. En este último caso se muestra también el número de grupos seleccionado por dicho criterio.

#	1ªEstrategia			2ªEstrategia			3ªEstrategia		
	25C	Tiempo de vida		25C	Tiempo de vida		25C	Tiempo de vida	
	Err	Err	Grupos	Err	Err	Grupos	Err	Err	Grupos
100	33	33	5	6	33	2	9	33	5
101	3	3	7	0	3	5	1	3	5
102	7	47	6	13	58	4	28	28	20
103	1	1	15	0	1	2	0	0	81
104	251	257	11	309	351	4	110	106	43
105	11	12	10	5	5	5	5	5	52
106	2	10	9	1	28	7	0	0	85
107	0	1	9	1	1	3	0	0	8
108	11	16	9	9	9	2	6	6	66
109	4	9	14	2	10	2	2	2	39
111	0	0	11	0	0	2	0	0	22
112	2	2	7	1	2	3	1	1	16
113	0	0	11	0	0	2	0	0	49
114	12	16	12	11	16	4	13	6	61
115	0	0	4	0	0	3	0	0	5
116	2	2	13	0	2	2	1	1	11
117	1	1	7	0	0	7	1	1	15
118	96	96	12	58	100	2	34	12	141
119	0	0	7	0	0	2	0	0	33
121	1	1	12	0	1	2	1	1	5
122	0	0	8	0	0	8	0	0	4
123	0	0	6	0	0	2	0	0	16
124	36	43	9	41	41	4	38	36	44
200	129	130	15	117	531	14	84	84	22

#	1ªEstrategia			2ªEstrategia			3ªEstrategia		
	25C	Tiempo de vida		25C	Tiempo de vida		25C	Tiempo de vida	
	Err	Err	Grupos	Err	Err	Grupos	Err	Err	Grupos
201	48	54	7	50	65	4	44	44	21
202	37	42	12	17	56	2	23	23	23
203	81	82	19	286	385	2	75	0	2977
205	14	14	9	13	15	6	13	15	16
207	187	196	20	52	318	5	133	17	64
208	107	109	17	120	449	3	96	98	21
209	181	298	5	106	162	3	66	65	31
210	32	37	17	30	71	2	33	26	57
212	0	0	11	3	4	4	1	0	43
213	112	351	8	90	396	3	126	126	29
214	6	6	14	4	5	12	3	3	73
215	4	5	20	9	26	3	4	4	17
217	34	50	10	65	69	10	4	3	47
219	11	12	9	11	18	3	11	11	18
220	94	94	3	4	94	2	30	94	2
221	1	3	8	1	1	6	0	0	41
222	389	389	10	328	421	2	338	0	2481
223	116	125	17	108	265	3	106	77	32
228	3	3	14	3	4	7	3	3	10
230	0	0	9	0	0	3	0	0	12
231	2	2	5	2	2	5	2	2	5
232	388	398	12	80	89	15	123	136	11
233	19	20	18	32	35	3	16	14	60
234	2	50	5	1	50	2	1	1	8
Total	2470	3020	508	1989	4192	203	1585	1087	6947
%	2.25	2.75		1.81	3.81		1.44	0.99	

Sobre los resultados de la Tabla 4.1 aplicamos test estadísticos para verificar si las diferencias entre las distintas estrategias son o no significativas. En primer lugar, se aplicó un test de normalidad Shapiro-Wilk [119] con el que se comprueba que no se cumple la hipótesis de normalidad de los resultados. Por ello se aplicó el test no paramétrico de

Tabla 4.2: P-valor del test Wilcoxon de significancia para las distintas estrategias a pares.

	1ªEst. vs 2ªEst.	1ªEst. vs 3ªEst.	2ªEst. vs 3ªEst.
25 Grupos	0.0821	<0.0001	0.0404
Criterio Tiempo Vida	0.0046	<0.0001	<0.0001

Wilcoxon a los resultados de las distintas estrategias [145]. En el caso de un número fijo de 25 grupos por registro se obtienen unos p-valores de 0.0821, < 0.0001 y 0.0404 para estrategia 1 vs. estrategia 2, estrategia 1 vs. estrategia 3 y estrategia 2 vs. estrategia 3, respectivamente. Con un nivel de significancia de 0.05 no se demuestra que la segunda estrategia sea mejor que la primera, pero la estrategia tercera, que emplea evidencia negativa, sí que es superior a la primera y a la segunda. En los resultados en los que se selecciona el número de grupos utilizando el criterio del tiempo de vida los p-valores fueron < 0.01 para todos los casos (véase Tabla 4.2), por lo que todos los resultados son significativamente diferentes con este criterio.

Se calculó la matriz de confusión para la tercera estrategia con 25 grupos. La matriz se obtuvo utilizando las anotaciones originales de la base de datos MIT-BIH Arrhythmia Database (véase Tabla 4.3) y con las anotaciones recomendadas por la AAMI (véase Tabla 4.4). En estas matrices las columnas representan el tipo real de los latidos según las anotaciones y las filas representan el tipo en que se agrupó.

4.2.3. Discusión

En los resultados con 25 grupos, a pesar de que aparenta haber una ligera mejora entre la primera y segunda estrategia, pasando de un error del 2.25 % a un error del 1.81 % ($p = 0.0821$), las diferencias encontradas no son estadísticamente significativas. Sin embargo, las diferencias entre la segunda y la tercera estrategia, con un error del 1.44 %, sí que lo son según los test aplicados ($p = 0.0404$). Esta mejora respecto a la segunda estrategia surge del cambio en el tratamiento de la información extraída de las distancias entre latidos, de la que se pasa a extraer evidencia negativa con el algoritmo PN-EAC. Estos resultados apoyan la asunción previa de que en algunos casos el uso de cierta información como evidencia negativa puede mejorar los resultados del agrupamiento basado en *ensembles*, demostrando así la utilidad de un algoritmo, como PN-EAC, que explota tanto evidencia positiva como evidencia negativa.

Al utilizar el criterio del tiempo de vida el error entre la primera y la segunda estrategia se incrementa del 2.75 % hasta el 3.81 %. Al mismo tiempo, el número total de grupos baja

Tabla 4.3: Matriz de confusión para la base de datos MIT-BIH Arrhythmia Database utilizando las anotaciones originales de la base de datos.

	N	**L**	**R**	**a**	**V**	**F**	**J**	**A**	**S**	**E**	**j**	**P**	**Q**	**!**	**e**	**f**
N	74920	0	1	9	125	106	4	395	2	1	177	0	11	0	16	11
L	0	8072	1	0	2	2	0	106	0	0	0	0	2	1	0	0
R	1	0	7212	0	0	3	29	118	0	2	5	0	0	0	0	0
a	2	0	0	132	1	1	0	3	0	0	0	0	0	0	0	0
V	26	0	0	4	6906	39	0	20	0	0	0	0	1	13	0	0
F	13	0	0	0	91	652	0	0	0	0	0	0	1	0	0	0
J	1	0	0	0	0	0	50	0	0	0	0	0	0	0	0	0
A	37	0	41	5	0	0	0	1897	0	0	2	0	0	0	0	0
S	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
E	0	0	1	0	0	0	0	0	0	101	0	0	0	1	0	0
j	14	0	0	0	0	0	0	5	0	0	45	0	0	0	0	0
P	0	0	0	0	1	0	0	0	0	0	0	6956	1	0	0	43
Q	0	0	0	0	0	0	0	0	0	0	0	0	5	0	0	0
!	0	0	0	0	4	0	0	0	0	2	0	0	0	457	0	0
e	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
f	2	0	0	0	0	0	0	0	0	0	0	68	12	0	0	928
Se	99.87	100.00	99.39	88.00	96.86	81.20	60.24	74.57	0.00	95.28	19.65	99.03	15.15	96.82	0.00	94.50
P+	98.87	98.61	97.86	94.96	98.53	86.13	98.04	95.71	-	98.06	70.31	99.36	100.00	98.70	-	91.88

Tabla 4.4: Matriz de confusión para la base de datos MIT-BIH Arrhythmia Database utilizando las anotaciones recomendadas por la AAMI.

	N	**S**	**V**	**F**	**Q**
N	90201	855	129	111	24
S	100	2147	3	1	0
V	28	22	7484	39	1
F	13	0	91	652	1
Q	2	0	1	0	8013
Se(%)	99.84	71.00	97.09	81.20	99.68
P+(%)	98.77	95.38	98.81	86.13	99.96

de 508 a 203, una media de 10.5 y 4.2 grupos por registro, respectivamente. Es deseable un número bajo de grupos, ya que esto significa que el cardiólogo tendrá que realizar menos trabajo, pero no a costa de un incremento del error. Cuando se utiliza en la tercera estrategia el error es 0.99 %. Pero en este caso el número de grupos se dispara, creando hasta un total de 6947 grupos (una media de 144.7 grupos por registro).

Un análisis del rendimiento del criterio del tiempo de vida basado únicamente en el número de grupos encontrados podría resultar bastante pobre. Resultaría más informativo conocer cuánto se separa el número de grupos encontrados con respecto al número de morfologías presentes en el registro. Una aproximación a esto se puede obtener comparando el número de grupos encontrados aplicando este criterio con el número de tipos distintos de anotaciones presentes en el registro (véase Tabla 1.1). Dicho cálculo da como resultado que el criterio del tiempo de vida encuentra, respectivamente, para la primera, segunda y tercera estrategias, 7.02, 0.66 y 141.15 grupos más por registro que tipos de anotaciones presentes. En la tercera estrategia, si descartamos los registros 203 y 222 del análisis, el número de grupos adicional por registro es de 28.84.

Esta diferencia de 7.02 grupos en la primera estrategia podría deberse a la confluencia de varias morfologías del latido bajo la misma etiqueta de clasificación. Las mayores diferencias se observan para los registros 213 y 209, que claramente se ven beneficiados de utilizar un número de grupos mayor que el seleccionado por el criterio. En otros registros como el 104 o el 220 no hay prácticamente ningún beneficio en utilizar un número de grupos mayor, lo que sugiere que habría acertado en su decisión.

Para la segunda estrategia el número de grupos encontrado es, a la vista de los resultados, menor al necesario, siendo en varios casos incluso inferior al número de tipos de anotaciones

presentes en el registro. Por ejemplo, en el registro 202 el criterio encuentra 2 grupos para representar los 5 tipos de anotaciones presentes, o en el registro 223 que se encuentran 3 grupos para representar los 6 tipos de anotaciones presentes.

El número de grupos encontrados para la tercera estrategia es claramente exacerbado, especialmente para los registros 203 y 222 en los que casi todos los grupos están formados por un solo latido. En otros registros como, por ejemplo, en el 118 se encuentran 141 grupos en un registro con 4 tipos de anotaciones. En algunos casos este incremento no va acompañado de una reducción en el error, como, por ejemplo, en el registro 108 en el que aun utilizando 66 grupos no se reduce el error con respecto al obtenido con 25 grupos. Adicionalmente, en algunos registros ya se obtenía una separabilidad perfecta con un menor número de grupos. Por ejemplo, en el registro 113 (2 tipos de anotaciones) con 25 grupos no había ningún error, por lo que parece innecesario incrementar dicho número hasta los 49 grupos encontrados por el criterio del tiempo de vida.

En general los resultados del criterio del tiempo de vida son insatisfactorios, debido a las diferencias en el número de grupos creado entre diferentes registros, y a lo elevado de este número en algunas ocasiones, especialmente en la tercera estrategia (hasta 2977 grupos). La proliferación de grupos, especialmente en algunos registros, se debe, al menos en parte, a la presencia de artefactos y ruido. Un registro atípico es el 231 en el que aparecen 5 tipos de anotaciones, y el criterio encuentra 5 grupos en las tres estrategias, obteniendo además, un error reducido.

En la Tabla 4.3 es posible apreciar que una parte considerable de los errores se cometen en los latidos auriculares prematuros (A), que son agrupados como latidos normales (N) o de bloqueo de rama (L y R). Los latidos auriculares prematuros se distinguen principalmente por una forma anormal de la onda P, por lo que no es de extrañar que PN-EAC, basada en el complejo QRS, tenga problemas para distinguir estos latidos. Otra fuente importante de errores son los latidos nodales de escape (j); en este caso de nuevo se diferencian por información aportada por la onda P y por la distancia entre latidos, que en PN-EAC estamos usando para separar latidos (evidencia negativa) pero no para agruparlos. Por ello muchos de estos latidos son agrupados como latidos normales. Sin embargo, para latidos en los que la morfología del complejo QRS cambia considerablemente, por ejemplo, en los latidos con bloqueo de rama izquierda (L), el rendimiento del algoritmo es más satisfactorio.

El uso de 16 funciones de Hermite permitiría el cálculo de la representación de Hermite en tiempo real, incluso sin el uso de una GPU, según los resultados del Capítulo 2. Además,

el uso de un mayor número de funciones de Hermite demostró no mejorar los resultados de agrupamiento, incluso empeorándolos en ocasiones. Al ser K-means un algoritmo voraz tiene dificultades inherentes para lidiar con una dimensionalidad alta del espacio de características. Aunque según BIC y AIC obtendríamos una representación óptima para un mayor porcentaje de latidos usando más coeficientes, una mayor exactitud en la representación no implica un mejor resultado en el agrupamiento. Una dimensionalidad muy elevada puede hacer que el resultado del agrupamiento empeore, pese a contar con un menor error de representación.

Es posible comparar nuestros resultados con los obtenidos por [77], donde también se usaron 25 grupos. El mejor resultado de agrupamiento obtenido por dicho trabajo tiene un error del 1.51 % para la base de datos MIT-BIH Arrhythmia Database completa. Comparado con los resultados aquí obtenidos utilizando un número fijo de 25 grupos (el mismo número empleado en [77]), la primera y segunda estrategia obtienen errores mayores (2.15 % y 1.81 % respectivamente). Sin embargo, la tercera estrategia, que incluye la evidencia negativa, obtiene un error de 1.44 %, ligeramente inferior al de [77]. Se realizará una comparativa más amplia con otros resultados de la bibliografía en el siguiente capítulo.

Los resultados obtenidos respaldan la idea inicial de que el agrupamiento mediante acumulación de evidencia puede ser una solución para el agrupamiento de latidos. La flexibilidad y adaptabilidad de este método le permite explorar información de las distintas derivaciones por separado, evitando así el problema de la maldición de la dimensionalidad [13][45]. Dado que cada derivación aporta matices diferentes sobre la actividad eléctrica del miocardio, es de esperar que añadir derivaciones adicionales mejore los resultados del agrupamiento. Esta idea será la explorada en la próxima sección.

4.3. Agrupamiento de latidos utilizando 12 derivaciones con PN-EAC

En la rutina médica a menudo se usan registros de hasta 12 derivaciones (véase Sección 1.1.3), y todas ellas son utilizadas por el cardiólogo para el diagnóstico [38]. En cambio, los trabajos de clasificación y agrupamiento automáticos de latidos típicamente solo utilizan una o dos derivaciones de los registros de ECG [30][31][77]. A esto ha contribuido el hecho de que la base de datos de referencia en identificación de arritmias, la MIT-BIH Arrhythmia Database (véase Sección 1.3.1), tiene únicamente dos derivaciones. No obstante, existen otras bases de datos anotadas con más derivaciones, como por ejemplo la St.-Petersburg Institute

of Cardiological Technics 12-lead Arrhythmia Database (INCARTDB) [103] (véase Sección 1.3.2).

Esta base de datos también goza de cierta popularidad en la comunidad científica. Sin embargo, incluso aquellos autores que la utilizan se limitan en su mayoría a seleccionar una o dos derivaciones. En [7] se realiza una clasificación de latidos utilizando árboles de decisión; para ello se utiliza la base de datos MIT-BIH Arrhythmia Database junto con 4000 latidos de la INCARTDB, pero usando una única derivación. En [94] se presenta un clasificador de latidos en dos etapas con un agrupamiento mediante el algoritmo Fuzzy C-means y clasificación utilizando redes neuronales. El trabajo se validó sobre la base de datos INCARTDB, entre otras, pero empleando solamente una derivación. Otro clasificador, en este caso basado en ELM (Extreme Learning Machines) que utiliza una derivación de esta base de datos es [6]. En [82] se utiliza la INCARTDB para evaluar la detección de latidos ventriculares prematuros, pero nuevamente utilizando una sola derivación. En [85] se utilizan dos derivaciones de esta base de datos para desarrollador un clasificador lineal de latidos. El mismo clasificador lineal desarrollado en [85] es utilizado en [83], pero en este caso es mejorado para ser capaz de utilizar las 12 derivaciones. El clasificador se aplica a todas las derivaciones, a un subconjunto de ellas y a la mejor derivación. Finalmente, en [84] se amplía el trabajo publicado en [83] realizando la validación sobre varias bases de datos, utilizando distintas representaciones del latido y con diferentes estrategias para combinar la información de las derivaciones.

El principal motivo por el que no se suelen usar más de un par de derivaciones es el incremento en la dimensionalidad del espacio de características empleado en la representación del latido. El uso de 12 derivaciones de modo concurrente significa multiplicar por 12 la dimensionalidad del espacio de características comparado con el uso de una única derivación. Este incremento en un orden de magnitud supone un reto en la aplicación de técnicas de aprendizaje automático. Sin embargo, empleando el paradigma de acumulación de evidencia cada derivación puede emplearse como una fuente complementaria de evidencia a combinar para la creación de la partición final, lo que elimina el problema del incremento en la dimensión del espacio de características y a la vez permite explotar y combinar toda la información de las 12 derivaciones.

En esta sección se empleará PN-EAC para crear un algoritmo de agrupamiento de latidos que trabaje de modo simultáneo sobre 12 derivaciones. También se estudiará cómo evolucionan los resultados del agrupamiento según se va incorporando información proveniente de un mayor número de derivaciones. Para ello probaremos a ejecutar el

algoritmo sobre 2, 4, 6, 8, 10 y 12 derivaciones y analizaremos los resultados para comprobar si existe una mejoría al incrementar el número de derivaciones.

En las pruebas realizadas en este capítulo se utilizaron 71 registros de los 75 de la base de datos INCARTDB (véase Sección 1.3.2). Los registros excluidos fueron el I02, I03, I57 y I58 debido a que en ellos una de las derivaciones está ausente, por lo que se prefirió excluirlos y trabajar solo sobre los registros que contienen las 12 derivaciones.

4.3.1. Estrategias para la generación de particiones

Se obtendrá para cada una de las derivaciones la representación de Hermite (coeficientes y σ), utilizando de nuevo 16 funciones. Posteriormente se calcularán características derivadas de la distancia entre latidos dadas por las ecuaciones (4.6) y (4.7) calculadas sobre las anotaciones de la base de datos. En consecuencia, cada latido será caracterizado por doce representaciones de Hermite (16 coeficientes y σ), una por cada derivación, y las dos características derivadas de la distancia entre latidos, en total 206 características por latido.

Esta dimensión del espacio de características puede resultar demasiado elevada para la mayoría de los algoritmos de aprendizaje automático, requiriendo generalmente el uso de técnicas de selección de características o de técnicas de reducción de dimensiones [113].

Anteriormente se propusieron tres estrategias para aplicar PN-EAC sobre registros de dos derivaciones (véase Sección 4.2.1). De entre ellas la estrategia basada en obtener evidencia positiva de cada representación de Hermite del latido y evidencia negativa de las características dadas por las ecuaciones (4.6) y (4.7) (Estrategia 3), fue la que obtuvo un mejor resultado (véase Tabla 4.1). Por tanto, nos basamos en esta estrategia para realizar la acumulación de evidencia.

Se realizaron pruebas utilizando 2, 4, 6, 8, 10 y 12 derivaciones, extrayendo siempre evidencia de cada derivación de modo independiente. Cada latido x_i, con d derivaciones, se representará de la siguiente forma:

$$\begin{aligned} v_i^1 &= (C^1, \sigma^1), \\ v_i^2 &= (C^2, \sigma^2), \\ &\dots \\ v_i^d &= (C^d, \sigma^d), \\ v_i^{d+1} &= (R_1, R_2). \end{aligned} \tag{4.13}$$

Al aplicar el algoritmo sobre 2 derivaciones el procedimiento es análogo al ya presentado en la sección anterior (véase Algoritmo 7); para cada una de las dos derivaciones se generan 100 particiones de las que se extraerá evidencia positiva. Estas particiones, al igual que el resto, se generan mediante el algoritmo K-means. En este caso debemos elegir 2 derivaciones de las 12 disponibles para generar las particiones. Esta elección se realiza de modo aleatorio. Se generan también 100 particiones con las características dadas por (4.6) y (4.7), de las que se extrae evidencia negativa. Todo el proceso se repite 100 veces para obtener una medida del error promedio de usar 2 derivaciones.

Para ejecutar PN-EAC con 4 derivaciones se procederá de forma similar. El primer paso es elegir aleatoriamente 4 de las 12 derivaciones disponibles. Para cada una de estas derivaciones se generan 100 particiones con la información de Hermite (coeficientes y σ). Por consiguiente, habrá en total 400 particiones de las que extraer evidencia positiva. Para mantener el mismo peso relativo de evidencia positiva con respecto a la evidencia negativa que el empleado anteriormente (1/2), se generan 200 particiones con las características derivadas de la distancia entre latidos ((4.6) y (4.7)). De este modo se evita que la información procedente de la distancia entre latidos se vaya diluyendo al incrementar el número de derivaciones empleadas para el agrupamiento. Evitar esta dilución es importante ya que la información derivada de las características de Hermite y la información derivada de la distancia entre latidos es complementaria. Para los casos de 6, 8, 10 y 12 derivaciones se procederá de forma análoga, obteniendo 100 particiones por cada derivación para generar evidencia positiva y ajustando el número de particiones generadas con las características dadas por las ecuaciones (4.6) y (4.7) para que el número de particiones que proporcionan evidencia negativa sea siempre 1/2 del número total de particiones que proporcionan evidencia positiva.

Para cada número de particiones repetiremos el proceso 100 veces. En cada ocasión las derivaciones utilizadas se eligen aleatoriamente entre las 12 disponibles; excepto en el caso de 12 derivaciones, donde se utilizan todas. La generación de particiones y la extracción de la partición final de la matriz de acumulación de evidencia se realizarán empleando el Algoritmo 7, modificando el número de derivaciones y el peso relativo de la evidencia negativa. En este caso no utilizaremos el criterio del tiempo de vida para seleccionar el número de grupos debido al mal rendimiento que mostró, en especial en lo relativo a la proliferación de grupos; los resultados se obtendrán siempre con 25 grupos.

4.3.2. Resultados

La Tabla 4.5 muestra el promedio de errores para los distintos números de derivaciones. En todos los casos el número de errores se obtiene utilizando un número fijo de 25 grupos por registro. El porcentaje de error se obtiene de dividir el número total de errores por el número total de latidos de la base de datos (165514 latidos, unos 2300 por registro). Los resultados medios de este porcentaje son 0.6006 %, 0.4052 %, 0.381 %, 0.3579 %, 0.3493 % y 0.3379 % para 2, 4, 6, 8, 10 y 12 derivaciones, respectivamente.

Tabla 4.5. Resultados del agrupamiento para 2 (d2), 4 (d4), 6 (d6), 8 (d8), 10 (d10) y 12 (d12) derivaciones. Se muestra para cada registro la media del número de errores en las 100 ejecuciones realizadas. Al final se muestra el promedio del número total de errores y el tanto por ciento de error sobre todos los latidos de la base de datos.

	d2	d4	d6	d8	d10	d12
I01	0.75	0.07	0	0	0	0
I04	33.79	31.87	30.42	28.58	28.42	27.89
I05	15.73	13.66	13.24	12.73	11.38	10.87
I06	23.35	13.84	10.05	9.44	9.47	9
I07	19.17	5.22	3.72	3.6	3.33	3.07
I08	6.35	3.64	3.76	3.13	3	2.37
I09	9.76	9.51	9.16	9.15	8.67	8
I10	0.85	0.2	0.01	0.01	0	0
I11	24.02	24	24	24	24	24
I12	3.97	2.99	3.11	2.96	3.05	3.15
I13	0	0	0	0	0	0
I14	0	0	0	0	0	0
I15	0.04	0	0	0	0	0
I16	0.04	0	0	0	0	0
I17	0.19	0.01	0	0	0	0
I18	57.08	52.74	51.52	50.54	50.37	51.7
I19	1.77	1.5	1.43	1.35	1.51	1.82
I20	102.1	60.59	52.78	47.6	47	39.87
I21	51.95	26.31	25.4	24.29	22.59	22.28
I22	63.79	41.17	37.23	36.81	35.93	37.31
I23	0.03	0	0	0	0	0

	d2	d4	d6	d8	d10	d12
I24	0.08	0	0	0	0	0
I25	1.57	1.05	1.14	1	1	1
I26	3.8	2.22	2.22	1.96	1.98	2
I27	0.01	0	0	0	0	0
I28	0	0	0	0	0	0
I29	10.55	2.09	1.23	0.61	0.35	0.12
I30	1.9	0.41	0.05	0	0	0
I31	41.17	35.25	45.28	33.29	29.44	19.85
I32	0.06	0	0	0	0	0
I33	90.59	26.28	16.7	15.36	15.07	15.3
I34	63.73	15.93	11.89	11.51	11.85	11.93
I35	59.05	45.68	41.64	37.67	34.94	34.96
I36	47.48	38.37	34.69	36.3	36.67	42.15
I37	1.02	1	1	1	1	1
I38	2.37	0.87	0.64	0.73	0.57	0.33
I39	1.68	0.15	0.03	0.01	0	0
I40	4.26	3.42	3.46	3.23	3.18	3.17
I41	4.96	4.95	4.94	4.82	4.93	5
I42	5.98	5.64	5.19	5.33	5.51	5.18
I43	7.05	5.96	5.07	5.01	5	5
I44	7.04	5.75	5.68	5.48	4.82	4.17
I45	0.02	0	0	0	0	0
I46	7.88	6.02	5.75	5.39	4.97	4.97
I47	2.86	1.12	1.07	1.03	1.02	1
I48	3.15	2.53	2.43	2.66	2.78	3
I49	0	0	0	0	0	0
I50	1.91	1.85	1.72	1.82	1.83	1.12
I51	3.34	3.09	2.88	3.01	2.93	3
I52	0	0	0	0	0	0
I53	0.1	0.03	0	0	0	0
I54	2.85	0.31	0.16	0.11	0.02	0
I55	1.47	0.45	0.09	0.04	0	0
I56	1.71	0.06	0.02	0	0	0
I59	38.17	37.62	37.83	34.87	33.96	33.05
I60	50.23	48.86	48.58	47.21	45.28	42.41

	d2	d4	d6	d8	d10	d12
I61	0.88	0.89	0.89	0.93	0.98	1
I62	36.72	26.77	25.6	24.22	23.49	21.92
I63	12.42	10.68	9.72	8.74	9.4	9.48
I64	2.43	1.94	2.2	1.93	1.96	2
I65	11.6	9.47	8.84	8.84	8.94	8.98
I66	1.81	1.77	1.91	1.85	1.81	2
I67	6.15	5.05	5	5	5	5
I68	1.37	1.01	1	1	1	1
I69	0.71	0.37	0.66	0.58	0.76	0.83
I70	0.01	0	0	0	0	0
I71	9.08	6.49	5.97	5.85	5.72	5.93
I72	7.77	7.46	7.56	7.54	7.86	8
I73	3.93	2.02	2.51	2.02	2	2
I74	15.11	12.37	11.66	10.35	11.46	10.04
I75	1.42	0.17	0.03	0	0	0
Total	994.2	670.7	630.8	592.5	578.2	559.2
%	0.601	0.405	0.381	0.358	0.349	0.338

Sobre los resultados se aplicaron test estadísticos para verificar la significancia de las diferencias entre utilizar un mayor o menor número de derivaciones [119]. Se rechazó la hipótesis de normalidad de los datos por lo que se aplicaron test no paramétricos. Primeramente, se aplicó el test de Friedman [46]. Dicho test toma como hipótesis nula que los resultados obtenidos con las distintas cantidades de derivaciones son iguales. Esta hipótesis fue rechazada ($p-valor < 0.001$), demostrando por tanto que hay diferencias entre los resultados obtenidos si variamos el número de derivaciones utilizado. Adicionalmente se compararon las distintas opciones de dos en dos, para ver si había diferencias significativas al incrementar el número de derivaciones. Para ello se utilizó el test Wilcoxon [145]. En todos los casos posibles se obtuvo un p-valor menor que 0.001, rechazando siempre la hipótesis nula. Por tanto, hay diferencias significativas en todos los casos al variar el número de derivaciones

Posteriormente a la aplicación del test de Friedman, si la hipótesis nula es rechazada, es posible aplicar otros test para obtener un ranking de los números de derivaciones que obtienen

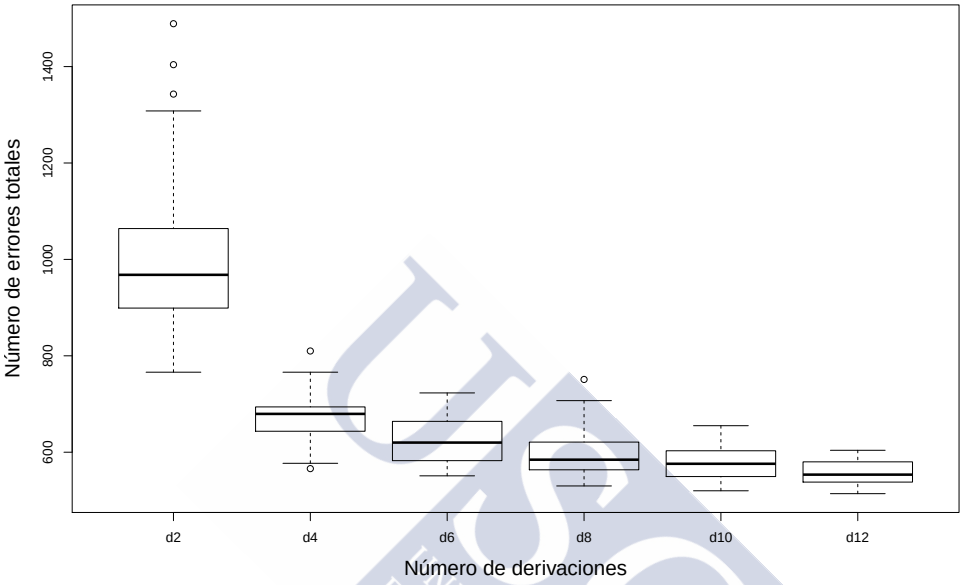


Figura 4.6: Resultados del error en toda la base de datos para los distintos números de derivaciones. Los puntos fuera de rango se representan como puntos individuales, las líneas verticales o "bigotes" son proporcionales a la diferencia entre cuartiles y las cajas van desde el valor del primer cuartil Q1 hasta el valor del tercer cuartil Q3, correspondiendo la línea horizontal intermedia con la mediana (Q2).

un mejor resultado. Para ello se aplicó el test Nemenyi [100]. Los resultados del test dieron el siguiente orden (de mejor a peor): 12, 10, 8, 6, 4 y 2 derivaciones. También se aplicaron otros test similares (Holm [59] y Finner [41]) obteniendo la misma ordenación.

4.3.3. Discusión

La Tabla 4.5 muestra que el error tiende a reducirse cuando se incrementa el número de derivaciones utilizadas. El error se reduce del 0.601 % de media usando 2 derivaciones hasta el 0.338 % utilizando las 12 derivaciones. Este decremento se puede apreciar en la Figura 4.6, donde se ve que el mayor cambio se produce al pasar de 2 a 4 derivaciones, mientras que al seguir añadiendo derivaciones las mejoras son menos pronunciadas.

También se observa que disminuye la variación en los resultados (rango intercuartílico) al añadir más derivaciones. Al revisar las cien ejecuciones realizadas para cada número de derivaciones, se aprecian diferencias en los resultados según las derivaciones concretas que se utilicen (I, II, etc.). Esto es especialmente notable al trabajar con cantidades más pequeñas de derivaciones. Por ejemplo, cuando se utilizan 2 derivaciones, algunas parejas de derivaciones obtienen un error de tan solo 860 latidos en toda la base de datos, mientras que otras parejas llegan hasta los 1400. Esta variabilidad disminuye conforme se aumenta el número de derivaciones utilizadas. Esto se debe a que en algunas de las derivaciones la señal tiene baja calidad (véase Figura 4.7). La evidencia extraída de estas derivaciones es pobre, lo que hace empeorar los resultados del agrupamiento. Cuando se emplea un número elevado de derivaciones el impacto relativo causado por la mala calidad en alguna de las derivaciones es compensado más fácilmente por el resto de las derivaciones. Por ello los resultados son más consistentes y disminuye la variabilidad.

Los resultados obtenidos sustentan la hipótesis inicial de que el agrupamiento basado en acumulación de evidencia puede mejorar al utilizar más derivaciones. Lo cual no implica que siempre se obtenga una mejora al utilizar un número superior de derivaciones. Por ejemplo, todas las ejecuciones que emplearon 4 derivaciones mejoran cualquier resultado utilizando 2 derivaciones. Sin embargo, algunas ejecuciones con 10 derivaciones obtienen un mejor resultado que algunas con 12 derivaciones (aunque en promedio se obtienen mejores resultados con 12 derivaciones; véase Figura 4.6). Algunas derivaciones en determinados casos pueden no aportan información nueva e incluso pueden empeorar el resultado añadiendo ruido al agrupamiento (véase Figura 4.7). Sin embargo, no es sencillo seleccionar a priori las mejores derivaciones para agrupar. Esto depende en muchas ocasiones del registro, de los tipos de latidos presentes o de las condiciones de grabación del registro. Por lo que, si no se cuenta con un método fiable para evaluar la calidad de una derivación y decidir si utilizarla o no en el agrupamiento, la mejor opción para aumentar el rendimiento y la robustez del agrupamiento, según los resultados obtenidos, es aumentar el número de derivaciones utilizadas.

En la bibliografía de agrupamiento de latidos no hemos encontrado ningún trabajo con el que comparar los resultados obtenidos en esta base de datos. Tampoco hemos encontrado ningún otro trabajo que aplique un algoritmo de agrupamiento a las doce derivaciones del ECG de modo simultáneo. Basándose en una técnica de agrupamiento es posible construir un clasificador asistido [29][109], con la ayuda de un cardiólogo que anote los grupos de latidos

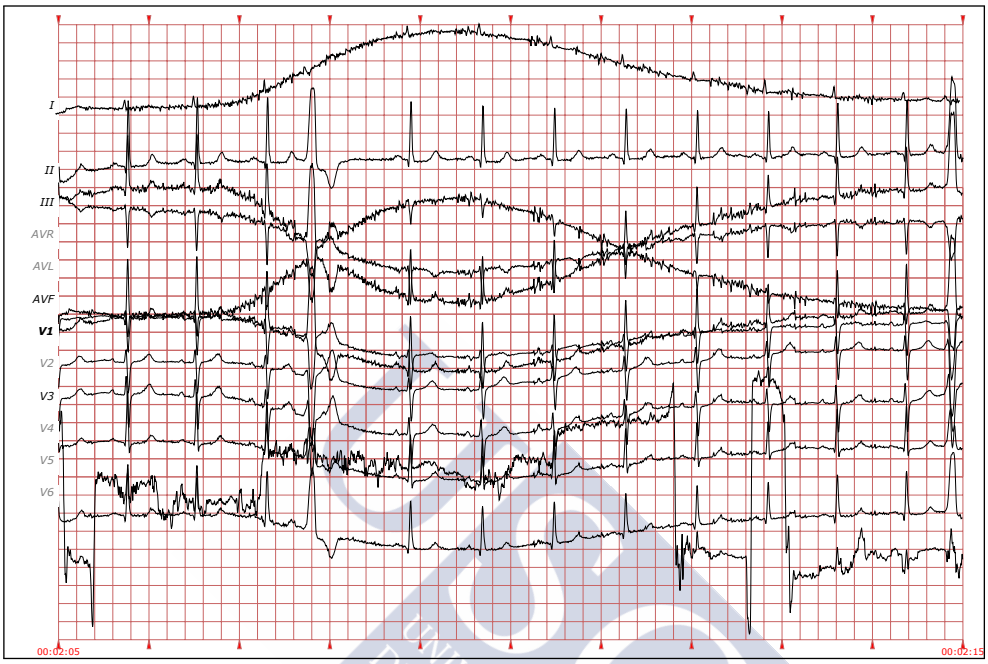


Figura 4.7: Fragmento de ECG de 12 derivaciones. Obsérvese cómo algunas derivaciones, I o V4, tienen mucho ruido y en ellas es difícil distinguir los latidos y la forma que tienen. (Fuente: St.-Petersburg Institute of Cardiological Technics 12-lead Arrhythmia Database, registro I66, entre 0:02:05 y 0:02:15)

(véase Sección 1.2). Los resultados de este clasificador asistido sí posibilitan una comparación con otros algoritmos de clasificación que fueron aplicados sobre la misma base de datos.

El clasificador propuesto en [6] obtiene, en el mejor caso, un error del 0.3 % sobre la base de datos INCARTDB, pero para obtener dicho error es necesario que se anoten al menos 200 latidos de cada registro. En este mismo trabajo también se muestra el error obtenido sin anotar ningún latido (permitiendo por tanto una comparación más directa con nuestro trabajo), siendo este del 23.6 %. En [85] se obtiene un error del 9.38 % en la base de datos INCARTDB. En [83], utilizando PCA sobre las 12 derivaciones y wavelets para representar los latidos, se obtuvo un error del 2.88 % para la base de datos INCARTDB completa, aunque dividiendo los latidos únicamente en tres clases (en nuestro caso se dividen en las 5 clases recomendadas por la AAMI).

Los resultados obtenidos avalan la idea de emplear el paradigma de acumulación de evidencia para la integración de información de múltiples derivaciones, así como el interés de usar evidencia negativa a partir de la información extraída de las posiciones de los latidos. Además, el algoritmo es fácilmente adaptable a cualquier número de derivaciones y altamente paralelizable (la evidencia de cada derivación puede obtenerse en paralelo con las demás).

La reducción del error al pasar de 2 (0.601 %) a 12 (0.338 %) derivaciones de casi el 50 % hace pensar que si en otras bases de datos, como la MIT-BIH Arrhythmia Database, estuvieran disponibles las 12 derivaciones el rendimiento de PN-EAC algoritmo podría mejorar de forma considerable.

USC
UNIVERSIDADE
DE SANTIAGO
DE COMPOSTELA

Capítulo 5

Agrupamiento dinámico de latidos mediante acumulación de evidencia

La técnica PN-EAC permite identificar grupos de latidos con formas y tamaños arbitrarios sin imponer un modelo predeterminado a los grupos, e integrar de forma sencilla información de múltiples derivaciones sin que el incremento en el número de dimensiones merme su rendimiento; todo lo contrario, cuantas más derivaciones estén disponibles mejor será el resultado del agrupamiento. Sin embargo, PN-EAC tiene dos limitaciones importantes. La primera es que requiere que estén disponibles todos los latidos que se van a agrupar. En cambio, cuando el ECG se obtiene mediante una monitorización en tiempo real (por ejemplo, un paciente ingresado en una UCI) en ocasiones es necesario proporcionar resultados parciales del agrupamiento según nuevos latidos van estando disponibles. La segunda limitación de esta técnica es su coste computacional, medido tanto en tiempo de CPU como en espacio de almacenamiento. Ambos escalan aproximadamente como $\mathcal{O}(n^2)$, siendo n el número de latidos a agrupar. En la práctica esto impide aplicar esta técnica sobre registros de larga duración, lo que hace que no sea adecuada para el análisis de registros Holter.

En la bibliografía de agrupamiento las técnicas dinámicas aparecen como una solución a ambos problemas [9][15][47][126]. Estas técnicas permiten procesar grandes conjuntos de datos de forma iterativa y proporcionar resultados de agrupamiento parciales a lo largo del tiempo. A pesar de estas ventajas, este tipo de técnicas tienen una presencia casi nula en la bibliografía de agrupamiento de latidos. Para realizar el procesado de registros con una gran

cantidad de latidos o para permitir proporcionar resultados intermedios la mayoría de los autores optan por dividir el registro en fragmentos que son procesados por separado [73][120]. Hasta donde alcanza nuestro conocimiento, solo en [20] se propone un algoritmo dinámico para el agrupamiento de latidos. En ese trabajo se supone que los latidos llegan como una secuencia ordenada de datos. La longitud de la secuencia de latidos es desconocida y, típicamente, demasiado elevada para permitir su almacenamiento completo en memoria principal. Por ello el algoritmo debe procesar cada latido en orden y una única vez, descartándolo a continuación.

En este capítulo se presenta una versión dinámica de la técnica PN-EAC que permite mostrar la evolución del agrupamiento, superando algunas de las limitaciones de dicha técnica. La versión dinámica permite proporcionar resultados parciales en cualquier instante de una monitorización en tiempo real y puede aplicarse sobre secuencias no acotadas de latidos, empleando una cantidad de memoria que puede llegar a ser varios órdenes de magnitud inferior a la requerida por PN-EAC.

5.1. Agrupamiento dinámico mediante acumulación de evidencia positiva y negativa (EPN-EAC)

Definimos $\mathscr{X}$ como una secuencia ordenada de objetos $\mathscr{X} = \{x_1, x_2, \dots\}$ en la que desconocemos el número total de objetos y en la cual el objeto x_i es el que ha llegado en la posición i. El tiempo de llegada de cada objeto consecutivo puede no ser equidistante. $\mathscr{X}$ es lo que en la literatura de agrupamiento dinámico se conoce como "flujo de datos" [2]. Denotamos por z el índice del último objeto disponible hasta el momento actual de la secuencia. En un instante determinado los objetos disponibles hasta x_z serán $\mathscr{X}_z = \{x_1, x_2, \dots, x_z\}$. El resultado de un algoritmo de agrupamiento aplicado sobre $\mathscr{X}_z$ será una partición $P^z = \{A_1^z, A_2^z, \dots, A_k^z\}$ que distribuye los z objetos disponibles en k grupos, donde A_j^z representa el grupo j de la partición P^z.

Definimos un *ensemble* sobre $\mathscr{X}_z$ como un conjunto de m particiones distintas $E^z = \{P_1^z, P_2^z, \dots, P_m^z\}$. Combinando todas estas particiones podemos obtener la partición final de los objetos hasta el objeto x_z, P_*^z.

5.1.1. Descripción de la técnica EPN-EAC

En este capítulo se propone una nueva técnica que adapta PN-EAC para hacerla dinámica e iterativa, y que denominaremos: agrupamiento dinámico mediante acumulación de evidencia positiva y negativa (*Evolving Positive and Negative Evidence Accumulation Clustering*, EPN-EAC). En EPN-EAC siempre se trabaja sobre una lista de objetos $W = \{w_1, w_2, ..., w_o\}$ de tamaño fijo o que representará a todos los objetos procesados hasta el objeto x_z. El tamaño de esta lista o es un parámetro del algoritmo que determina el tamaño de la matriz de evidencia y el número de objetos con los que se crearán las particiones. El tamaño o debe elegirse de acuerdo a los recursos computacionales disponibles y los requisitos de rendimiento, pero también debe ser lo suficientemente grande como para poder resumir adecuadamente $\mathscr{X}$.

Definimos Ω como una matriz de evidencia de tamaño $o \times o$ para la lista de objetos W. Cada celda de $\Omega_{(i,j)}$ representa la evidencia recolectada para una pareja de objetos w_i y w_j. Mientras el número de objetos z de $\mathscr{X}_z$ es menor que o, cada nuevo x_z es almacenado en W. Esta es una fase de inicialización del algoritmo de EPN-EAC (véase Algoritmo 8, línea 1). Cuando $z > o$, cada vez que llega un nuevo objeto es necesario hacer espacio en W para él. Para ello se seleccionan los dos objetos w_i y w_j más similares en W (véase Algoritmo 8, línea 3); que serán fusionados (véase Sección 5.1.4) para convertirse en un único objeto. En W se eliminan los dos objetos originales w_i y w_j y se añade el objeto resultado de su fusión, quedando por tanto un espacio libre en W para el nuevo objeto x_z. A continuación, se recopila evidencia sobre el conjunto W generando nuevas particiones de datos. Finalmente, de forma opcional, es posible extraer el resultado final del agrupamiento de todos los objetos procesados P_*^z. Cada una de las partes de este ciclo de ejecución se describirá detalladamente en las siguientes secciones.

Las sucesivas actualizaciones que EPN-EAC lleva a cabo sobre la lista W requieren de ciertas modificaciones sobre la técnica de acumulación de evidencia PN-EAC. El hecho de que los objetos cambien implica que en un punto de ejecución determinado no se habrán generado el mismo número de particiones para todos los objetos de W. Por tanto, es necesario contabilizar, además de la evidencia, cuántas veces han estado presentes cada par de objetos w_i y w_j en una misma partición de datos. Esta información es recogida en una nueva matriz Φ de tamaño $o \times o$. Cada celda (i, j) de la matriz Φ indica cuántas particiones de datos se han creado con los dos objetos w_i y w_j, mientras que la celda homóloga en la matriz Ω indica en cuántas de dichas particiones esos objetos se agruparon en el mismo grupo. Dividiendo las

Algoritmo 8 Esquema general de EPN-EAC

[tbp]

Input: Flujo de datos $\mathscr{X} = \{x_1, x_2, \dots\}$
Input: o tamaño de la lista de objetos W
Input: p número de particiones a generar en la inicialización
Input: r número de particiones a generar en cada iteración
Output: P_* partición final de los datos

1: (W, Ω, Φ, Y)=Inicialización$(\mathscr{X}, o, p)$ #Procedimiento 1
2: **while** nuevo objeto x_z **do**
3: (w_i, w_j)=BuscarPareja(W, Φ, Ω) #Procedimiento 2
4: (W, Y, Φ, Ω)= FusiónPareja$(W, Y, \Phi, \Omega, x_z, w_i, w_j)$ #Procedimiento 3
5: Generar r particiones con los objetos de W #Sección 5.1.5
6: Extraer evidencia de las particiones
7: (Opcional) (P_*^z)=ExtraerPartFinal(W, Y, Φ, Ω) #Procedimiento 4
8: **end while**

celdas de ambas matrices es posible obtener el porcentaje de veces que dos objetos fueron agrupados juntos en todas las particiones en las cuales ambos estuvieron presentes.

Cada iteración del bucle del algoritmo (véase Algoritmo 8, líneas 2-8) fusiona dos objetos de W y añade uno nuevo. Al final de cada iteración es posible obtener el resultado de la partición final de los objetos de W. Sin embargo, dado que $z > o$, a partir de dicha agrupación de los objetos de W no es posible obtener un agrupamiento de todos los objetos de $\mathscr{X}_z$. Para resolver este problema se vinculará cada objeto de $\mathscr{X}_z$ con su representante en W mediante un vector $Y = (y_1, y_2, \dots, y_z)$ de tamaño z, donde y_l es la posición en W del representante correspondiente; es decir, cada objeto $x_l \in \mathscr{X}_z$ estará representado por un $w_m \in W$ tal que $y_l = m$. De esta forma, aunque EPN-EAC realiza un agrupamiento sobre W, es posible obtener un agrupamiento de todos los objetos de $\mathscr{X}_z$. Para ello, en el resultado del agrupamiento P_*^W reemplazamos cada objeto w_m por todos los objetos $\{x_a, x_b, x_c, \dots\} \in \mathscr{X}_z$, tales que:

$$\forall x_i \in \{x_a, x_b, x_c, \dots\} : y_i = m; \tag{5.1}$$

obteniendo así P_*^z. Este vector Y debe ser actualizado cada vez que haya cambios en los objetos de W (al fusionar y al añadir objetos).

5.1.2. Inicialización del algoritmo

En la fase de inicialización del algoritmo se introduce en la lista W los primeros o objetos de $\mathscr{X}$, $W = \{w_1 = x_1, w_2 = x_2, \ldots, w_o = x_o\}$ (véase Procedimiento 1, línea 2). Con dichos objetos se generan p particiones diferentes de datos para extraer evidencia (véase Procedimiento 1, líneas 3-4). La forma de crear las particiones ya fue presentada anteriormente (véase Sección 4.1.1). De dichas particiones se recopila la evidencia al igual que en el caso estático y se almacena en la matriz Ω. Para la evidencia positiva sería:

$$\Omega_{(i,j)} = a_{ij}, \tag{5.2}$$

siendo a_{ij} el número de veces en que la pareja de objetos w_i, w_j han sido agrupados en el mismo grupo a lo largo de las p particiones.

Si procede, también se podría extraer evidencia negativa de otras particiones (véase Sección 4.1.2). Es importante que la cantidad de la evidencia negativa y positiva guarden una proporción adecuada al problema. Típicamente el número de particiones de evidencia positiva p deberían ser mayor que las de evidencia negativa q, que actúan a modo de restricciones sobre el agrupamiento propuesto por la primera. Esta evidencia se combinaría en la misma matriz:

$$\Omega_{(i,j)} = \Omega_{(i,j)} - b_{ij}, \tag{5.3}$$

siendo b_{ij} el número de veces en que la pareja de objetos w_i y w_j no aparece en el mismo grupo en las q particiones.

Procedimiento 1 Inicialización EPN-EAC

[bp]

Input: Flujo de datos $\mathscr{X} = \{x_1, x_2, \ldots\}$
Input: o tamaño de la lista de objetos W
Input: p número de particiones a generar en la inicialización
Output: W, Ω, Φ y Y

1: **Function** Inicialización($\mathscr{X}, o, p$)
2: Inicializar W con los primeros o objetos de $\mathscr{X}$, $W = \{w_1 = x_1, w_2 = x_2, \ldots, w_o = x_o\}$
3: Generar p particiones con los objetos en W #Sección 4.1.1
4: Extraer evidencia y acumularla en la matriz Ω #Ecuación (5.2)
5: Inicializar matriz Φ con el valor p
6: Inicializar Y, $Y = (y_1 = 1, y_2 = 2, \ldots, y_o = o)$
7: **return** (W, Ω, Φ, Y)
8: **end**

En la fase de inicialización todos los objetos aparecen en las p particiones iniciales, por lo que toda la matriz Φ se inicializará con este valor, $\Phi_{(:,:)} = p$. Únicamente es necesario almacenar el número de particiones de las que se extrae evidencia positiva ya que será este número el que indique el máximo posible de particiones en las que dos objetos se agruparon juntos.

Por último, también es necesario inicializar el vector Y, de tal modo que los o primeros objetos de W estén vinculados con sus correspondientes objetos de $\mathscr{X}$, $(y_1 = 1, y_2 = 2, \ldots, y_o = o)$. En este momento concluye el proceso de inicialización y comienza la ejecución del bucle (véase Algoritmo 8, líneas 2-8) que procesará cada nuevo objeto x_z dinámicamente.

5.1.3. Búsqueda del par de objetos más similares en W

Al recibir un objeto nuevo es necesario liberar espacio en W para almacenarlo. Para ello el primer paso es buscar la pareja de objetos (w_i, w_j) más similares (véase Procedimiento 2). Se comienza calculando la proporción de veces que cada par de objetos w_i y w_j fueron agrupados juntos mediante la división elemento a elemento de las matrices Ω y Φ:

$$\Psi_{(i,j)} = \frac{\Omega_{(i,j)}}{\Phi_{(i,j)}}. \tag{5.4}$$

La matriz Ψ proporciona una medida de similitud entre los objetos de W. A mayor valor de $\Psi_{(i,j)}$, mayor la similitud entre los correspondientes objetos w_i y w_j. Por ello, se buscará el máximo de la matriz Ψ y por tanto a las parejas de objetos más similares. Debemos identificar todas las celdas de la matriz donde aparece el máximo, ya que las correspondientes parejas de objetos tendrán la misma similitud. Dada la simetría de la matriz únicamente se considerarán las celdas por encima de la diagonal (véase Procedimiento 2, líneas 3-12):

$$I = \left\{ (i,j) \mid (i,j) = argmax_{l,m}(\Psi_{(l,m)}), l < m \right\}. \tag{5.5}$$

En el caso de que I solo contenga una pareja estos serán los objetos a fusionar y se procede al siguiente paso del algoritmo (véase Sección 5.1.4). En otro caso, es necesario determinar cuál de todas las parejas con el mismo valor $\Psi_{(i,j)}$ será fusionada. En el paso anterior se calculó la similitud de dos objetos basándose en si son agrupados juntos. Ahora se medirá de acuerdo a cómo es su agrupamiento con el resto de objetos. El hecho de que un objeto w_i tienda a agruparse con los mismos objetos con los que se agrupa w_j es un indicio de que w_i y w_j son similares.

Procedimiento 2 Búsqueda pareja de objetos (w_i, w_j) más similares en W

Input: Lista de objetos $W = w_1, w_2, \ldots, w_o$, Matrices Φ y Ω
Output: Pareja de objetos w_i y w_j

Function BuscarPareja(W, Φ, Ω)
  $\Psi = \Omega / \Phi$ (elemento a elemento) #Ecuación (5.4)
  $Max = -\infty$, $I = \{\}$
  for $m = 1$ **to** $o - 1$ **do**
    for $l = m + 1$ **to** o **do**
      if $\Psi_{(l,m)} > Max$ **then**
        $I = \{(w_l, w_m)\}, Max = \Psi_{(l,m)}$ #Ecuación (5.5)
      else
        if $\Psi_{(l,m)} == Max$ **then** $I = \{I, (w_l, w_m)\}$ **end if**
      end if
    end for
  end for
  if $size(I) == 1$ **then**
    $(w_i, w_j) = I$
  else
    $MinDistance = \infty$, $J = \{\}$
    for all $(w_l, w_m) \in I$ **do**
      if $distance(\Psi_{(l,:)}, \Psi_{(m,:)}) < MinDistance$ **then**
        $J = \{(w_l, w_m)\}, MinDistance = distance(\Psi_{(l,:)}, \Psi_{(m,:)})$ #Ecuación (5.6)
      else
        if $distance(\Psi_{(l,:)}, \Psi_{(m,:)}) == MinDistance$ **then** $J = \{J, (w_l, w_m)\}$ **end if**
      end if
    end for
    if $size(J) == 1$ **then**
      $(w_i, w_j) = J$
    else
      $MinDistance = \infty$,$K = \{\}$
      for all $(w_l, w_m) \in J$ **do**
        if $distance(w_l, w_m) < MinDistance$ **then**
          $K = \{(w_l, w_m)\}, MinDistance = distance(w_l, w_m)$ #Ecuación (5.7)
        else
          if $distance(w_l, w_m) == MinDistance$ **then** $K = \{K, (w_l, w_m)\}$ **end if**
        end if
      end for
      if $size(K) == 1$ **then** $(w_i, w_j) = K$ **else** $(w_i, w_j) = random.pair(K)$ **end if**
    end if
  end if
  return (w_i, w_j)
end

Cada columna o fila de la matriz Ψ representa el agrupamiento de un objeto con el resto de objetos de W. Por lo tanto, comparando dos filas o columnas se puede comparar el agrupamiento de dos objetos con respecto a todos los objetos en W. De esta forma se obtiene un nuevo criterio de similitud para encontrar aquellas parejas de objetos cuyo agrupamiento en W es más similar. Sobre las parejas (w_i, w_j) que cumplen el criterio anterior (5.5), elegimos solo aquellas que minimizan la distancia en la matriz Ψ entre las filas i y j (véase Procedimiento 2, líneas 16-23):

$$J = \left\{(i,j) \mid (i,j) = argmin_{l,m}\left(distancia(\Psi_{(l,:)}, \Psi_{(m,:)}\right), \forall (l,m) \in I\right\}. \tag{5.6}$$

En el caso de que en J solo haya una pareja de objetos se procede al siguiente paso del algoritmo. En caso contrario, se medirá la distancia euclídea entre los objetos de cada pareja restante. Aquella pareja con una menor distancia entre objetos será la elegida. Esto se aplica para todas las parejas de J, seleccionando aquella con menor distancia (véase Procedimiento 2, líneas 27-34):

$$K = \left\{(i,j) \mid (i,j) = argmin_{l,m}\left(distancia(w_l, w_m)\right), \forall (l,m) \in J\right\}. \tag{5.7}$$

Finalmente, en el caso de que en K haya más de una pareja se escogerá una pareja aleatoriamente (véase Procedimiento 2, línea 35).

5.1.4. Fusión de dos objetos e inclusión del nuevo objeto x_z

En el paso anterior se seleccionó la pareja (w_i, w_j) de objetos en W más similares según los criterios establecidos. En este paso se fusionan dichos objetos, dejando un espacio libre en W en el cual introducir el nuevo objeto x_z (véase Procedimiento 3).

El objeto resultado de la fusión de (w_i, w_j) se calcula como la media de los vectores de características de los dos objetos. El nuevo objeto resultante de dicha fusión ocupará la posición del antiguo objeto w_i en W:

$$w_i \leftarrow media(w_i, w_j), \tag{5.8}$$

donde $\leftarrow$ indica que w_i es reemplazado por la media de w_i y w_j.

A continuación, se actualizará Y de tal modo que todos los objetos que estaban representados por los dos objetos pasarán a estar representados por el nuevo objeto. Para ello en Y todos los objetos con valor j pasan a tener valor i, la posición del objeto fusionado (véase Procedimiento 3, líneas 3-5).

Procedimiento 3 Fusión de dos objetos w_i y w_j

Input: Lista de objetos $W = w_1, w_2, \ldots, w_o$, Y, Matrices Φ y Ω, nuevo objeto x_z
Input: Elementos a fusionar w_i y w_j
Output: W, Y, Φ y Ω actualizados

1: **Function** FusiónPareja$(W, Y, \Phi, \Omega, x_z, w_i, w_j)$
2: $w_i \leftarrow media(w_i, w_j)$ #Ecuación (5.8)
3: **for** $l = 1$ **to** $z-1$ **do**
4: **if** $y_l == j$ **then** $y_l = i$ **end if**
5: **end for**
6: **for** $l = 1$ **to** o **do**
7: $\Omega_{(l,i)} = \Omega_{(l,i)} + \Omega_{(l,j)}$, $\Phi_{(l,i)} = \Phi_{(l,i)} + \Phi_{(l,j)}$ #Ecuación (5.10)
8: $\Omega_{(i,l)} = \Omega_{(l,i)}$, $\Phi_{(i,l)} = \Phi_{(l,i)}$
9: **end for**
10: $w_j \leftarrow x_z$
11: $y_z = j$
12: **for** $l = 1$ **to** o **do**
13: $\Omega_{(j,l)} = 0$, $\Omega_{(l,j)} = 0$, $\Phi_{(j,l)} = 0$, $\Phi_{(l,j)} = 0$
14: **end for**
15: **return** (W, Y, Φ, Ω)
16: **end**

También precisarán actualización las matrices Ω y Φ para mantener la consistencia con el conjunto W que representan. La evidencia de los dos antiguos objetos será asignada al objeto fusión (véase Procedimiento 3, líneas 6-9):

$$\left((\Omega_{(:,i)})^T = \Omega_{(i,:)} \right) \leftarrow \Omega_{(i,:)} + \Omega_{(j,:)}, \tag{5.9}$$

$$\left((\Phi_{(:,i)})^T = \Phi_{(i,:)} \right) \leftarrow \Phi_{(i,:)} + \Phi_{(j,:)}. \tag{5.10}$$

Posteriormente, se añade el nuevo objeto x_z al espacio libre correspondiente al antiguo objeto w_j, $w_j \leftarrow x_z$, Actualizando consecuentemente Y, de modo que $y_z = j$. El objeto w_j es nuevo y no hay ninguna evidencia sobre él. En consecuencia, las correspondientes columnas y filas de Ω y Φ son puestas a cero ($\Omega_{(:,j)} = 0$, $\Omega_{(j,:)} = 0$, $\Phi_{(:,j)} = 0$ y $\Phi_{(j,:)} = 0$).

5.1.5. Recolección de evidencia

Tras añadir el nuevo objeto x_z es preciso recopilar evidencia para el conjunto actualizado W, (para los objetos antiguos tenemos evidencia ya recolectada, pero para el nuevo no hay

ninguna). Para recolectar la evidencia se crean r particiones nuevas con los objetos de W. La elección de r debe realizarse cuidadosamente ya que este número de particiones es creado cada vez que se recibe un nuevo objeto, por lo que influirá notablemente en el rendimiento del algoritmo. Un valor de r elevado puede acarrear un coste computacional excesivo mientras que uno demasiado bajo puede impedir recopilar suficiente evidencia para agrupar correctamente el nuevo objeto.

El proceso de recolección de evidencia es similar al utilizado en la inicialización. De las r particiones creadas se extrae evidencia que se añade a la ya obtenida mediante la expresión:

$$\Omega_{(i,j)} = \Omega_{(i,j)} + a_{ij}, \tag{5.11}$$

donde a_{ij} es la cantidad de veces que la pareja de objetos w_i, w_j ha aparecido en el mismo grupo en las r particiones creadas. Al igual que en la inicialización, se podrían crear opcionalmente al mismo tiempo s particiones para extraer evidencia negativa:

$$\Omega_{(i,j)} = \Omega_{(i,j)} - b_{ij}, \tag{5.12}$$

siendo b_{ij} el número de veces en que la pareja de objetos w_i, w_j no aparece en el mismo grupo a lo largo de las s particiones.

También se debe actualizar la matriz Φ para reflejar las r nuevas particiones de evidencia positiva creadas:

$$\Phi_{(i,j)} = \Phi_{(i,j)} + r. \tag{5.13}$$

De este modo se han generado con el nuevo objeto de W r particiones de las que se habrá extraído evidencia.

5.1.6. Extracción de la partición final

Este último paso se ejecuta únicamente cuando sea necesario conocer un resultado parcial del agrupamiento de los objetos procesados hasta un momento dado (véase Procedimiento 4). Toda la información de evidencia acumulada hasta el momento está contenida en las dos matrices Ω y Φ, que serán utilizadas para obtener la partición final.

El primer paso para obtener la partición final es calcular Ψ, según (5.4). Con esta operación se obtiene una matriz de similitud sobre la que aplicar un algoritmo de agrupamiento diseñado para matrices de similitud. En nuestro caso se aplica el algoritmo jerárquico de enlace medio, tal y como hicimos en la acumulación de evidencia estática

Procedimiento 4 Extraer la partición final P_*^z

Input: Lista de objetos $W = w_1, w_2, \ldots, w_o$, Y, Matrices Φ y Ω
Output: P_*^z partición final de los datos

```
1:  Function ExtraerPartFinal(W, Y, Φ, Ω)
2:      Ψ = Ω/Φ (elemento a elemento)  #Ecuación (5.4)
3:      P*^W = enlace.medio(Ψ)  #Sección 4.1.3
4:      P*^z = P*^W
5:      for all (w_l) ∈ P*^W do
6:          L = {}
7:          for m = 1 to z do
8:              if y_m == l then L = {L, (x_m)} end if
9:          end for
10:         P*^z.replace(w_l).by(L)  #Ecuación (5.1)
11:     end for
12:     return (P*^z)
13: end
```

(véase Procedimiento 4, líneas 2-3). Como resultado obtenemos el agrupamiento final P_*^W de los objetos del conjunto W. Si se quiere extender dicho agrupamiento a todos los objetos de $\mathscr{X}_z$ se puede hacer mediante el array Y (véase Procedimiento 4, líneas 5-11). Para ello cada objeto de W será reemplazado por los objetos correspondientes de $\mathscr{X}_z$ (véase Ecuación (5.1)). Con este último paso finalizaría la iteración del algoritmo y quedaría en espera de un nuevo objeto para procesar.

5.1.7. Ejemplo ilustrativo del funcionamiento del algoritmo

Para ilustrar el funcionamiento de la técnica se presenta un ejemplo de su ejecución paso a paso. Dada una serie de datos $\mathscr{X} = \{11, 14, 18, 49, 3, 94, \ldots\}$, elegimos como tamaño de la lista W, $o = 4$. El número de particiones generadas en la inicialización será $p = 10$ y las particiones que se generan en cada iteración será $r = 5$. Solo se generará evidencia positiva.

Inicialización

El conjunto W es inicializado con los primeros o objetos de $\mathscr{X}$, $W = \{11, 14, 18, 49\}$. Con dichos objetos generamos $p = 10$ particiones de datos de las cuales extraemos evidencia (véase Ecuación (5.2)). Supongamos que la matriz de evidencia Ω obtenida es:

$$\Omega = \begin{bmatrix} 10 & 5 & 4 & 2 \\ 5 & 10 & 5 & 2 \\ 4 & 5 & 10 & 2 \\ 2 & 2 & 2 & 10 \end{bmatrix}.$$

Inicializamos la matriz Φ con el número de particiones:

$$\Phi = \begin{bmatrix} 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 \end{bmatrix}.$$

También se inicializa el array Y con los primeros o objetos de $\mathscr{X}$ vinculados, respectivamente, a los primeros o objetos de W:

$$Y = (1,2,3,4).$$

Primera iteración

Llega un nuevo objeto a procesar, $x_5 = 3$. El primer paso es encontrar la pareja de objetos más similares en W. Para ello obtenemos la matriz Ψ aplicando (5.4):

$$\Psi = \begin{bmatrix} 1 & 0.5 & 0.4 & 0.2 \\ 0.5 & 1 & 0.5 & 0.2 \\ 0.4 & 0.5 & 1 & 0.2 \\ 0.2 & 0.2 & 0.2 & 1 \end{bmatrix}.$$

Seleccionamos las celdas con el valor máximo sobre la matriz triangular superior, como se indica en (5.5). De las celdas con valor máximo obtenemos las parejas de índices que presentan similitud máxima:

$$I = \{(1,2),(2,3)\}.$$

Dado que hay más de una pareja en I, medimos la distancia entre las correspondientes columnas o filas de la matriz Ψ, y seleccionamos la pareja (o parejas) con una distancia menor (véase Ecuación (5.6)):

$$\begin{aligned} distancia(\Psi_{(1,:)}, \Psi_{(2,:)}) &= \sqrt{(1-0.5)^2 + (0.5-1)^2 + (0.4-0.5)^2 + (0.2-0.2)^2} \\ &= \sqrt{0.51}, \\ distancia(\Psi_{(2,:)}, \Psi_{(3,:)}) &= \sqrt{(0.5-0.4)^2 + (1-0.5)^2 + (0.5-1)^2 + (0.2-0.2)^2} \\ &= \sqrt{0.51}, \\ J &= \{(1,2),(2,3)\}. \end{aligned}$$

Utilizando este criterio ambas parejas obtienen el mismo valor de similitud, lo que no nos permite elegir una de ellas. Por tanto, procederemos a medir la distancia euclídea entre los objetos de W que representan las parejas de J, quedándonos con la pareja con menor distancia (véase Ecuación (5.7)):

$$\begin{aligned} distancia(w_1, w_2) &= \sqrt{(11-14)^2} = \sqrt{9} = 3, \\ distancia(w_2, w_3) &= \sqrt{(14-18)^2} = \sqrt{16} = 4, \\ K &= \{(1,2)\}. \end{aligned}$$

La pareja con menor distancia será la de los objetos w_1 y w_2, que procederemos a fusionar aplicando (5.8):

$$\begin{aligned} w_1 &\leftarrow media(w_1, w_2) = media(11, 14) = 12.5, \\ W &= \{12.5, 14, 18, 49\}. \end{aligned}$$

Actualizaremos Y para indicar que los objetos x_1 y x_2 ahora son representados por w_1:

$$Y = (1, 1, 3, 4).$$

Realizaremos las actualizaciones necesarias también en las matrices Ω y Φ para fusionar la evidencia de los dos antiguos objetos, como se indica en (5.10):

$$
\begin{aligned}
(\Omega_{(:,1)})^T &= \Omega_{(1,:)} \leftarrow \Omega_{(1,:)} + \Omega_{(2,:)} \\
&= \{10+5, 5+10, 4+5, 2+2\} = \{15, 15, 9, 4\}, \\
(\Phi_{(:,1)})^T &= \Phi_{(1,:)} \leftarrow \Phi_{(1,:)} + \Phi_{(2,:)} \\
&= \{10+10, 10+10, 10+10, 10+10\} = \{20, 20, 20, 20\},
\end{aligned}
$$

$$
\Omega = \begin{bmatrix} 15 & 15 & 9 & 4 \\ 15 & 10 & 5 & 2 \\ 9 & 5 & 10 & 2 \\ 4 & 2 & 2 & 10 \end{bmatrix}, \quad \Phi = \begin{bmatrix} 20 & 20 & 20 & 20 \\ 20 & 10 & 10 & 10 \\ 20 & 10 & 10 & 10 \\ 20 & 10 & 10 & 10 \end{bmatrix}.
$$

En el espacio libre dejado en W introduciremos el nuevo objeto $x_5 = 3$, $w_2 \leftarrow x_5$. Al ser un objeto nuevo actualizamos las matrices de evidencia, poniendo a cero las filas y columnas correspondientes a w_2:

$$
W = \{12.5, 3, 18, 49\},
$$

$$
\Omega = \begin{bmatrix} 15 & 0 & 9 & 4 \\ 0 & 0 & 0 & 0 \\ 9 & 0 & 10 & 2 \\ 4 & 0 & 2 & 10 \end{bmatrix}, \quad \Phi = \begin{bmatrix} 20 & 0 & 20 & 20 \\ 0 & 0 & 0 & 0 \\ 20 & 0 & 10 & 10 \\ 20 & 0 & 10 & 10 \end{bmatrix}.
$$

Se actualiza Y para indicar que el objeto x_5 está representado en W por el objeto w_2:

$$
Y = (1, 1, 3, 4, 2).
$$

Con el conjunto W actualizado generaremos $r = 5$ particiones de datos nuevas. Obtendremos la evidencia de dichas particiones y la sumaremos a la ya existente aplicando (5.11) y (5.13):

$$
\Omega = \begin{bmatrix} 15 & 0 & 9 & 4 \\ 0 & 0 & 0 & 0 \\ 9 & 0 & 10 & 2 \\ 4 & 0 & 2 & 10 \end{bmatrix} + \begin{bmatrix} 5 & 1 & 4 & 1 \\ 1 & 5 & 2 & 1 \\ 4 & 2 & 5 & 1 \\ 1 & 1 & 1 & 5 \end{bmatrix} = \begin{bmatrix} 20 & 1 & 13 & 5 \\ 1 & 5 & 2 & 1 \\ 13 & 2 & 15 & 3 \\ 5 & 1 & 3 & 15 \end{bmatrix}.
$$

$$
\Phi = \begin{bmatrix} 20 & 0 & 20 & 20 \\ 0 & 0 & 0 & 0 \\ 20 & 0 & 10 & 10 \\ 20 & 0 & 10 & 10 \end{bmatrix} + \begin{bmatrix} 5 & 5 & 5 & 5 \\ 5 & 5 & 5 & 5 \\ 5 & 5 & 5 & 5 \\ 5 & 5 & 5 & 5 \end{bmatrix} = \begin{bmatrix} 25 & 5 & 25 & 25 \\ 5 & 5 & 5 & 5 \\ 25 & 5 & 15 & 15 \\ 25 & 5 & 15 & 15 \end{bmatrix}.
$$

Para finalizar la iteración podríamos extraer la partición final de los datos si lo deseáramos. Para ello en primer lugar crearíamos la matriz Ψ sobre la que aplicaríamos el algoritmo de agrupamiento correspondiente para obtener P_*^W:

$$\Psi = \begin{bmatrix} 0.8 & 0.2 & 0.52 & 0.2 \\ 0.2 & 1 & 0.4 & 0.2 \\ 0.52 & 0.4 & 1 & 0.2 \\ 0.2 & 0.2 & 0.2 & 1 \end{bmatrix},$$

$$P_*^W = \{(w_1, w_3), (w_2), (w_4)\}.$$

Por medio de Y podríamos trasladar este agrupamiento al conjunto $\mathscr{X}$ (véase (5.1):

$$P_*^W = \{(w_1, w_3), (w_2), (w_4)\}.$$
$$Y = (1,1,3,4,2),$$
$$w_m \leftarrow \{x_l\} \forall x_l \mid y_l = m,$$
$$P_*^{\mathscr{X}} = \{(x_1, x_2, x_3), (x_5), (x_4)\}.$$

Segunda Iteración

Llega un nuevo objeto $x_6 = 94$. Aplicamos (5.4):

$$\Psi = \begin{bmatrix} 0.8 & 0.2 & 0.52 & 0.2 \\ 0.2 & 1 & 0.4 & 0.2 \\ 0.52 & 0.4 & 1 & 0.2 \\ 0.2 & 0.2 & 0.2 & 1 \end{bmatrix}.$$

Seguidamente (5.5):

$$I = \{(1,3)\}.$$

Al tener solo una pareja de objetos en I podemos fusionarlos aplicando (5.8):

$$w_1 \leftarrow media(w_1, w_3) = media(12.5, 18) = 15.25,$$
$$W = \{15.25, 3, 18, 49\}.$$

Actualizamos Y:

$$Y = (1,1,1,4,2).$$

Aplicamos (5.10):

$$\begin{aligned}(\Omega_{(:,1)})^T &= \Omega_{(1,:)} \leftarrow \Omega_{(1,:)} + \Omega_{(3,:)} \\ &= \{20+13, 1+2, 13+15, 5+3\} = \{33,3,28,8\}, \\ (\Phi_{(:,1)})^T &= \Phi_{(1,:)} \leftarrow \Phi_{(1,:)} + \Phi_{(3,:)} \\ &= \{25+25, 5+5, 25+15, 25+15\} = \{50,10,40,40\},\end{aligned}$$

$$\Omega = \begin{bmatrix} 33 & 3 & 28 & 8 \\ 3 & 5 & 2 & 1 \\ 28 & 2 & 15 & 3 \\ 8 & 1 & 3 & 15 \end{bmatrix}, \quad \Phi = \begin{bmatrix} 50 & 10 & 40 & 40 \\ 10 & 5 & 5 & 5 \\ 40 & 5 & 15 & 15 \\ 40 & 5 & 15 & 15 \end{bmatrix}.$$

Introducimos x_6 en W, $w_3 \leftarrow x_6 = 94$. Ponemos a cero las correspondientes filas y columnas:

$$W = \{12.5, 3, 94, 49\},$$

$$\Omega = \begin{bmatrix} 33 & 3 & 0 & 8 \\ 3 & 5 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 8 & 1 & 0 & 15 \end{bmatrix}, \quad \Phi = \begin{bmatrix} 50 & 10 & 0 & 40 \\ 10 & 5 & 0 & 5 \\ 0 & 0 & 0 & 0 \\ 40 & 5 & 0 & 15 \end{bmatrix}.$$

Actualizamos Y :

$$Y = (1,1,1,4,2,3).$$

Con el nuevo objeto, generamos r nuevas particiones de las que extraemos evidencia aplicando (5.11) y (5.13):

$$\Omega = \begin{bmatrix} 33 & 3 & 0 & 8 \\ 3 & 5 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 8 & 1 & 0 & 15 \end{bmatrix} + \begin{bmatrix} 5 & 3 & 1 & 2 \\ 3 & 5 & 0 & 1 \\ 1 & 0 & 5 & 1 \\ 2 & 1 & 1 & 5 \end{bmatrix} = \begin{bmatrix} 38 & 6 & 1 & 10 \\ 6 & 10 & 0 & 2 \\ 1 & 0 & 5 & 1 \\ 10 & 2 & 1 & 20 \end{bmatrix}.$$

$$\Phi = \begin{bmatrix} 50 & 10 & 0 & 40 \\ 10 & 5 & 0 & 5 \\ 0 & 0 & 0 & 0 \\ 40 & 5 & 0 & 15 \end{bmatrix} + \begin{bmatrix} 5 & 5 & 5 & 5 \\ 5 & 5 & 5 & 5 \\ 5 & 5 & 5 & 5 \\ 5 & 5 & 5 & 5 \end{bmatrix} = \begin{bmatrix} 55 & 15 & 5 & 45 \\ 15 & 10 & 5 & 10 \\ 5 & 5 & 5 & 5 \\ 45 & 10 & 5 & 20 \end{bmatrix}.$$

Para finalizar, si lo deseamos extraemos P_*^W:

$$\Psi = \begin{bmatrix} 0.69 & 0.4 & 0.2 & 0.22 \\ 0.4 & 1 & 0 & 0.2 \\ 0.2 & 0 & 1 & 0.2 \\ 0.22 & 0.2 & 0.2 & 1 \end{bmatrix},$$

$$P_*^W = \{(w_1, w_2), (w_3, w_4)\}.$$

Y podemos mapear P_*^W a $\mathscr{X}$ por medio de Y, aplicando (5.1):

$$\begin{aligned} P_*^W &= \{(w_1, w_2), (w_3, w_4)\}, \\ Y &= (1,1,1,4,2,3), \\ w_m &\leftarrow \{x_l\} \forall x_l \mid y_l = m, \\ P_*^{\mathscr{X}} &= \{(x_1, x_2, x_3, x_5), (x_4, x_6)\}. \end{aligned}$$

5.1.8. Análisis de complejidad computacional

Se analizó la complejidad computacional de EPN-EAC, al igual que se hizo para PN-EAC. En este caso el coste de inicialización del algoritmo es fijo y no depende de n (el número de objetos total) por lo que no se incluye en el cálculo. La parte principal del algoritmo es un bucle que se ejecuta cada vez que se recibe un nuevo elemento. Dicho bucle busca la pareja de elementos más similares en W (de tamaño o), fusiona la pareja, en el espacio libre añade el elemento nuevo y finalmente genera particiones sobre W y extrae evidencia (véase Algoritmo 8, líneas 3-6).

Las operaciones de encontrar la pareja de elementos más similares, fusionarla e incluir el elemento nuevo implican recorrer varias veces matrices de tamaño $o \times o$, y por lo tanto tienen una complejidad $\mathscr{O}(o^2)$. La complejidad de K-means es la misma que en PN-EAC, pero en este caso depende de tamaño o, siendo $\mathscr{O}(o \cdot \sqrt{o})$. Dado que se ejecutará r veces por cada iteración la complejidad total de la ejecución de K-means es $\mathscr{O}(r \cdot o \cdot \sqrt{o})$. Posteriormente es necesario extraer la evidencia, lo que tendrá una complejidad de $\mathscr{O}(o^2)$ por cada partición. Por tanto, la complejidad de la extracción de evidencia es $\mathscr{O}(r \cdot o^2)$.

La complejidad de la extracción de la partición final por parte del algoritmo jerárquico tiene la misma forma que en PN-EAC, pero en este caso dependiendo de o en vez de n, siendo $\mathscr{O}(o^2 \cdot log(o))$. Al igual que en PN-EAC este término podría llegar a ser superior a la complejidad de la extracción de evidencia ($\mathscr{O}(r \cdot o^2)$) pero en muchos casos se cumplirá que

$r >> log(n)$. Además, se debe tener en cuenta que este es un paso opcional en el algoritmo, que no es necesario ejecutar en cada iteración.

Por ello, consideraremos que en la mayoría de escenarios la complejidad total del bucle será:

$$\mathcal{O}(o^2) + \mathcal{O}(r \cdot o \cdot \sqrt{o}) + \mathcal{O}(r \cdot o^2) + \mathcal{O}(o^2 \cdot log(o)) \approx \mathcal{O}(r \cdot o^2). \tag{5.14}$$

El bucle debe ejecutarse por cada elemento, por lo que la complejidad total de las n iteraciones de EPN-EAC es $\mathcal{O}(n \cdot r \cdot o^2)$. Esta complejidad, para valores grandes de n y siempre que se cumpla $o << n$, es mucho menor que la de PN-EAC ($\mathcal{O}(p \cdot n^2)$). En cuanto a la complejidad espacial vendrá delimitada por las matrices necesarias, que en este caso, de tamaño $o \times o$, será $\mathcal{O}(o^2)$.

5.2. Aplicación al agrupamiento de latidos

En esta sección se presenta una aplicación de la técnica EPN-EAC al problema del agrupamiento de latidos. Para ello procederemos de forma análoga a como se hizo en el Capítulo 4, en donde se aplicó la técnica de acumulación de evidencia estática a la base de datos MIT-BIH Arrhythmia Database. Ello permitirá una comparación directa de los resultados con los obtenidos mediante PN-EAC. En este caso también se utilizarán las tres estrategias propuestas en la Sección 4.2.1, con pequeñas modificaciones para su uso en agrupamiento dinámico.

En primer lugar, es necesario definir el tamaño o de W, de tal modo que sea lo suficientemente grande como para representar todos los latidos del registro, pero que al mismo tiempo permita obtener una mejora significativa en el rendimiento. Tras varias pruebas se estableció experimentalmente un valor de $o = 100$ para cada uno de los registros de la base de datos MIT-BIH Arrhythmia Database. Un valor mayor de o no mejora significativamente los resultados, e incrementa el coste computacional. Teniendo en cuenta que cada registro de la base de datos MIT-BIH Arrhythmia Database consta de alrededor de 2000 latidos, emplear $o = 100$ implica una reducción de los requerimientos de memoria de aproximadamente $(2000^2/100^2) = 400$, ya que pasamos de una matriz de evidencia de tamaño $n \times n$ a una matriz de tamaño $o \times o$.

También es necesario fijar el número de particiones que se generan, tanto para la inicialización como en cada iteración para cada estrategia. Dichos valores se eligieron

intentado replicar la configuración seguida en el capítulo anterior, de modo que se generan 300 particiones en la inicialización y 30 particiones con cada nuevo latido.

Por consiguiente, las estrategias presentadas en el capítulo anterior se adaptan de la siguiente forma:

Estrategia 1: se generan 300 particiones al inicio con todas las características y con cada nuevo latido se generan otras 30 nuevas particiones con todas las características.

Estrategia 2: en la inicialización se generan 100 particiones por cada derivación de la base de datos (200 entre ambas) y otras 100 particiones con la información dada por las características derivadas de la distancia entre latidos (véanse Ecuaciones (4.6) y (4.7)) (sumando 300 particiones en total). Posteriormente, para cada latido nuevo se generan 10 particiones por cada derivación y 10 con la información derivada de las distancias entre latidos (30 en total). En todas las particiones se utilizará únicamente evidencia positiva.

Estrategia 3: al igual que en la anterior se generan 100 particiones por cada derivación, pero en esta estrategia generamos 100 particiones para evidencia negativa con la información de ritmo. De esta forma respetamos que la evidencia negativa sea 1/2 de las particiones de evidencia positiva, al igual que en el capítulo anterior. Con cada nueva iteración procedemos de forma similar: se generan 10 particiones por cada derivación para evidencia positiva y 10 particiones para evidencia negativa con la información de ritmo.

Con estos parámetros se intenta reproducir lo más fielmente posible las condiciones del experimento en el Capítulo 4, de forma que sea posible realizar una comparación directa y apreciar el impacto que la estrategia dinámica tiene en el resultado.

Tabla 5.1. Resultados del agrupamiento dinámico para las tres estrategias. Se muestra, para cada estrategia, el número de errores (Err.) utilizando un número fijo de 25 grupos (25C) y utilizando el criterio del tiempo de vida. En este último caso se muestra también el número de grupos seleccionado por dicho criterio.

#	**1ªEstrategia**			**2ªEstrategia**			**3ªEstrategia**		
	25C	TiempoVida		25C	TiempoVida		25C	TiempoVida	
	Err.	Err.	Grupos	Err.	Err.	Grupos	Err.	Err.	Grupos
100	33	33	3	0	33	3	0	0	99
101	3	3	5	0	5	2	0	1	9
102	36	38	8	28	155	2	28	28	11
103	2	2	8	1	0	99	0	0	99
104	257	368	7	181	98	99	202	111	99
105	10	11	10	5	2	99	5	2	99
106	2	3	7	0	2	6	0	0	99
107	0	0	6	0	0	6	0	0	6
108	9	16	11	7	2	99	5	2	99
109	0	1	12	0	0	99	0	0	28
111	0	0	9	0	0	2	0	0	99
112	2	2	2	0	2	2	0	2	2
113	1	1	3	0	0	3	0	0	3
114	12	16	4	6	16	4	6	4	99
115	0	0	7	0	0	99	0	0	99
116	2	2	7	0	2	3	0	2	4
117	1	1	5	0	1	6	0	1	6
118	95	96	7	29	96	4	34	5	99
119	0	0	3	0	0	2	0	0	26
121	1	1	8	0	1	2	0	0	99
122	0	0	4	0	0	3	0	0	99
123	0	0	4	0	0	4	0	0	99
124	39	39	13	38	33	99	36	33	99
200	149	151	9	43	25	99	50	27	99
201	52	53	7	42	32	99	45	35	99
202	36	40	7	20	12	99	17	11	99
203	102	103	6	105	66	99	69	63	99

#	1ªEstrategia			2ªEstrategia			3ªEstrategia		
	25C	TiempoVida		25C	TiempoVida		25C	TiempoVida	
	Err.	Err.	Grupos	Err.	Err.	Grupos	Err.	Err.	Grupos
205	14	14	11	12	19	2	12	10	99
207	322	140	99	255	21	99	236	54	99
208	67	74	14	90	77	99	90	73	99
209	382	383	7	38	23	99	33	24	99
210	39	55	11	28	14	99	34	14	99
212	0	0	6	0	0	99	0	0	99
213	162	162	11	221	112	99	182	163	99
214	2	23	11	1	24	3	1	1	99
215	4	5	10	4	2	99	4	2	99
217	7	36	13	3	1	99	1	0	99
219	8	13	8	8	7	99	8	7	99
220	94	94	2	3	94	2	4	3	99
221	0	2	6	0	0	99	0	0	99
222	421	421	4	312	180	99	295	187	99
223	230	372	8	286	40	99	66	37	99
228	3	3	10	3	3	8	3	3	12
230	0	0	4	0	0	4	0	0	99
231	2	2	5	2	2	5	2	2	5
232	394	398	4	73	71	99	64	48	99
233	16	18	13	13	5	99	15	6	99
234	50	50	4	4	50	2	0	0	4
Total	3061	3245	443	1861	1328	2555	1547	961	3680
%	2.78	2.95		1.69	1.21		1.41	0.87	

5.3. Resultados

En la Tabla 5.1 es posible ver los resultados del agrupamiento dinámico. Se muestran los errores en cada registro de la base de datos MIT-BIH Arrhythmia Database para cada una de las tres estrategias usando los dos criterios de selección del número de grupos (el criterio de

Tabla 5.2: P-valor del test Wilcoxon de significancia para las distintas estrategias dinámicas a pares.

	1ªEst. vs 2ªEst.	1ªEst. vs 3ªEst.	2ªEst. vs 3ªEst.
25 Grupos	0.0002	<0.0001	0.1449
Criterio Tiempo Vida	0.0002	<0.0001	0.0575

Tabla 5.3: P-valor del test Wilcoxon de significancia para cada estrategia entre la versión estática y la versión dinámica.

	1ªEstrategia	2ªEstrategia	3ªEstrategia
25 Grupos	0.0408	0.0115	0.0039
Criterio Tiempo Vida	0.9033	<0.0001	0.0124

tiempo de vida y un número fijo de 25 grupos). En este caso también se dividió el número de errores por el número de latidos totales (109966 latidos) para obtener el porcentaje de error.

Para los resultados con el número fijo de 25 grupos se obtuvieron unos porcentajes de error de 2.78 %, 1.69 % y 1.41 % para la primera, segunda y tercera estrategias, respectivamente; mientras que en el caso del criterio del tiempo de vida los porcentajes son de 2.95 %, 1.21 % y 0.87 %, respectivamente, teniendo en cuenta que en este caso el número de grupos creado es variable, con una media de 9.22, 53.22 y 76.66 grupos por registro para la primera, segunda y tercera estrategia, respectivamente.

Sobre los resultados de la Tabla 5.1, al igual que en la estrategia estática, se aplicaron test estadísticos para verificar la significancia de las diferencias entre las distintas estrategias. Los resultados obtenidos por el test de Wilcoxon son mostrados en la Tabla 5.2.

También se aplicaron test estadísticos para verificar la significancia de las diferencias en cada estrategia entre la versión estática y la versión dinámica. Los resultados se muestran en la Tabla 5.3.

Además, para la tercera estrategia y un número de grupos fijo de 25 (error del 1.41 %), se calculó también la matriz de confusión. Se eligió esta estrategia por ser la que obtuvo un mejor resultado con un número de grupos por registro reducido. Dicha matriz de confusión se muestra con las anotaciones originales de la base de datos MIT-BIH Arrhythmia Database y con las anotaciones recomendadas por la AAMI en las Tablas 5.4 y 5.5, respectivamente. En estas matrices las columnas representan el tipo real de los latidos según las anotaciones y las filas el tipo en que se agrupó.

Tabla 5.4: Matriz de confusión para la base de datos MIT-BIH Arrhythmia Database utilizando las anotaciones originales de la base de datos.

	N	**L**	**R**	**a**	**V**	**F**	**J**	**A**	**S**	**E**	**j**	**P**	**Q**	**!**	**e**	**f**
N	74781	0	1	9	70	199	4	175	2	0	104	0	10	0	14	18
L	0	8068	0	0	0	0	0	0	0	0	0	0	1	0	0	0
R	0	0	7185	0	5	1	29	29	0	2	5	0	0	7	0	0
a	0	0	0	135	4	0	0	7	0	0	0	0	0	0	0	0
V	33	0	0	3	7018	82	0	0	0	1	0	0	2	6	0	0
F	10	0	0	0	22	521	0	0	0	0	1	0	0	0	0	0
J	0	0	0	0	0	0	50	0	0	0	0	0	0	0	0	0
A	117	0	70	3	1	0	0	2227	0	0	0	0	0	0	2	0
S	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
E	0	0	0	0	1	0	0	0	0	103	0	0	0	97	0	0
j	72	0	0	0	0	0	0	0	0	0	119	0	0	0	0	0
P	0	0	0	0	0	0	0	0	0	0	0	6873	0	0	0	39
Q	0	0	0	0	0	0	0	0	0	0	0	0	5	0	0	1
!	0	4	0	0	8	0	0	106	0	0	0	0	0	362	0	0
e	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
f	3	0	0	0	1	0	0	0	0	0	0	151	15	0	0	924
Se	99.69	99.95	99.02	90.00	98.43	64.88	60.24	87.54	0.00	97.17	51.97	97.85	15.15	76.69	0.00	94.09
P+	99.20	99.99	98.93	92.47	98.22	94.04	100.00	92.02	-	51.24	62.30	99.44	83.33	75.42	-	84.46

Tabla 5.5: Matriz de confusión para la base de datos MIT-BIH Arrhythmia Database utilizando las anotaciones recomendadas por la AAMI.

	N	**S**	**V**	**F**	**Q**
N	90035	371	84	200	29
S	259	2543	5	0	0
V	37	109	7596	82	2
F	10	1	22	521	0
Q	3	0	1	0	8008
Se(%)	99.66	84.09	98.55	64.88	99.61
P+(%)	99.25	90.59	97.06	94.04	99.95

5.4. Discusión

Es posible apreciar en los resultados con 25 grupos una mejora significativa entre la primera estrategia y la segunda estrategia, pasando del 2.78 % al 1.69 % ($p = 0.0002$). De la segunda estrategia a la tercera se produce otra mejora del 1.69 % al 1.41 %, pero no es significativa estadísticamente ($p = 0.1449$). Ocurre algo semejante al utilizar el criterio del tiempo de vida, siendo significativa la diferencia entre la primera y la segunda estrategia del 2.95 % al 1.69 % ($p = 0.0002$), mientras que la diferencia entre la segunda y tercera estrategia no supera el umbral de 0.05 ($p = 0.0575$).

Los resultados son similares a los obtenidos anteriormente para el agrupamiento estático (véase Sección 4.2.2), reforzando las conclusiones ya alcanzadas en dicho capítulo con la técnica PN-EAC. En esta ocasión se aprecia una mayor diferencia entre la primera estrategia y la segunda, al contrario que en la versión estática donde las diferencias más significativas se obtenían entre la segunda y tercera estrategias.

Por otra parte, se corrobora, una vez más, el pobre comportamiento del criterio del tiempo de vida en la segunda y tercera estrategias. En este caso se estableció un límite superior de 99 grupos por registro que fue alcanzado en un alto número de registros en la segunda estrategia y por la mayoría en la tercera estrategia. Por ello de nuevo se reafirma la no idoneidad de este criterio para el problema del agrupamiento de latidos, especialmente al combinar evidencia de distintas representaciones e incluir evidencia negativa. Por ello, el resto del análisis lo basaremos sobre los resultados obtenidos utilizando 25 grupos.

Comparando los resultados con los obtenidos en el Capítulo 4 es posible apreciar que la adaptación de la técnica (de estática a dinámica) no parece haber empeorado el resultado en

términos de error. De hecho, los resultados para la segunda y tercera estrategias han mejorado ligeramente. En la técnica estática (PN-EAC) obteníamos porcentajes de error del 2.25 %, 1.81 % y 1.44 % para la primera, segunda y tercera estrategias, respectivamente, mientras que para la versión dinámica (EPN-EAC) se obtienen porcentajes de error del 2.78 %, 1.69 % y 1.41 %, siendo significativas todas estas diferencias entre las versiones estáticas y dinámicas significativas (véase Tabla 5.3). Esta ligera mejora para la segunda y tercera estrategias se debe principalmente a que en la versión dinámica recopilamos evidencia muchas más veces, incluso si la hacemos sobre un conjunto de latidos más pequeño.

Es posible comparar los resultados mostrados en la Tabla 5.4 con los obtenidos por [77] y los obtenidos por [20]. En esta comparación podemos ver cómo EPN-EAC obtiene mejor sensibilidad en 7 (L, a, V, J, E, Q, f) de las 14 clases (clases S y e no fueron consideradas al no estar suficientemente representadas), siendo mejores [20] y [77] en 5 (R, F, A, j, P) y 2 (N, !) de las clases, respectivamente. En cuanto al valor predictivo positivo, EPN-EAC obtuvo el mejor resultado para 7 (N, L, R, V, F, J, P) clases, mientras que [20] y [77] lo obtuvieron para 3 (A, !, f) y 4 (a, E, j, Q) clases, respectivamente. Finalmente, el error total obtenido es del 1.41 %, ligeramente inferior a los obtenidos por [20] y [77] de 1.44 % y 1.50 %, respectivamente.

Los peores resultados se obtienen para los latidos de fusión de marcapasos (F), los latidos prematuros supraventriculares (S) y atrioventriculares (J), los latidos de escape auricular (e), y los latidos inclasificables (Q), con los cuales el algoritmo se comporta de forma poco satisfactoria. El tipo F corresponde a la fusión de latidos de marcapasos y latidos normales, una distinción que es inherentemente difícil de hacer, incluso para los especialistas (véase Figura 5.1). Durante la ejecución, más de un 22 % de los latidos de este tipo fueron agrupados incorrectamente, principalmente como latidos normales (N). Los latidos de escape auricular (e) y los latidos prematuros supraventriculares (S), auriculares (A) y atrioventriculares (J), además de estar muy poco representados en la base de datos, tienen una morfología muy similar a la de los latidos normales, por lo que resulta muy difícil para un algoritmo distinguirlos (véanse Figuras 4.3 , 5.2 y 5.3). Por otra parte, el tipo Q aglutina a aquellos latidos que en la base de datos no se han podido clasificar en ninguna de las categorías, debido a una morfología no reconocida o al ruido de la señal, por lo que se trata de una clase sin una característica morfológica reconocible.

Los mejores resultados se obtienen para los latidos con bloqueo de rama izquierda (L) y para los latidos ventriculares (V), obteniendo en ambos casos mejor sensibilidad y valor predictivo positivo que los otros dos algoritmos. En ambos tipos de latido la morfología es

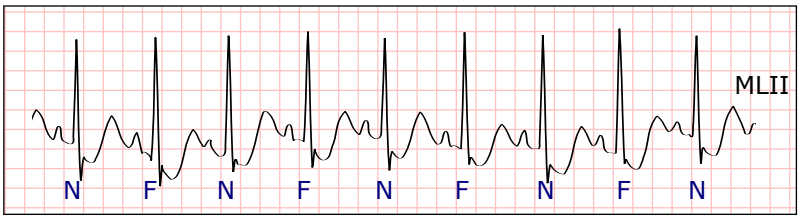


Figura 5.1: Fragmento de un ECG en el que se alternan latidos normales con latidos del tipo F. (Fuente: MIT-BIH Arrhythmia Database, registro 213, entre 0:02:07 y 0:02:12)

claramente distinta a la del complejo QRS normal y asimétrica, por lo que la representación elegida es especialmente adecuada para distinguir estos tipos de latido. Además, ambos grupos de latidos están entre los más numerosos, después de los latidos normales. En general, los tres trabajos, al estar basados en la morfología del complejo QRS, muestran un rendimiento más pobre al distinguir latidos con una morfología similar entre sí, que para distinguirse requieren de información de la onda P o, en su defecto, información derivada de la distancia entre latidos. El algoritmo aquí presentando podría verse beneficiado de incorporar información de la onda P como forma de evidencia negativa, de forma que ayudara a separar estos casos y al mismo tiempo el resultado del agrupamiento no se viera afectado por la inestabilidad de su detección.

En [77] con un número fijo de 25 grupos por registro se obtuvo un error del 1.51 % para la base de datos MIT-BIH Arrhythmia Database completa. Los resultados de este capítulo con la primera y segunda estrategias obtienen porcentajes de error mayores, mientras que la tercera estrategia obtiene un error ligeramente menor (1.41 %). Los resultados de sensibilidad obtenidos son algo inferiores a los obtenidos por [77], pero se compensa con un mejor resultado en el valor predictivo positivo.

Al comparar los resultados con los obtenidos por [20] es posible ver una gran semejanza, variando muy poco el resultado del error total sobre la base de datos MIT-BIH Arrhythmia Database, del 1.44 % obtenido por [20] al 1.41 % obtenido aquí. Los valores de sensibilidad obtenidos por [20] son ligeramente superiores, mientras que los de valor predictivo positivo son ligeramente inferiores. Es necesario recalcar que en [20] no se utiliza un número fijo de grupos de latidos, sino que dicho número se decide dinámicamente para cada registro. Sin embargo, a diferencia de la técnica aquí presentada, la información derivada de la distancia entre latidos no está integrada en el algoritmo principal de agrupamiento, sino que se aplica

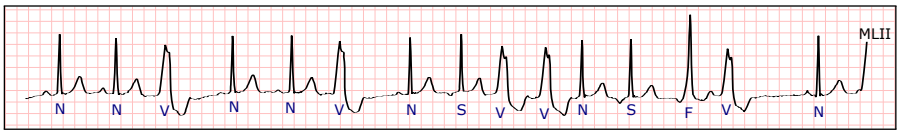


Figura 5.2: Fragmento de un ECG en el que se puede apreciar como los tipos de latido S y N son casi idénticos en la morfología del complejo QRS, distinguiéndose únicamente por tener la onda P invertida. (Fuente: MIT-BIH Arrhythmia Database, registro 208, entre 0:17:45 y 0:17:55)

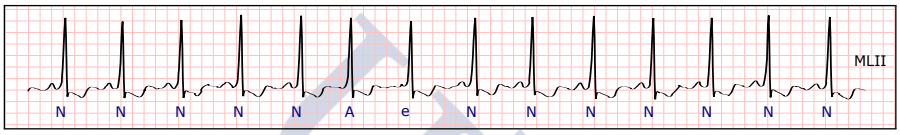


Figura 5.3: Fragmento de un ECG con latidos del tipo e y A entre latidos normales (N). (Fuente: MIT-BIH Arrhythmia Database, registro 223, entre 0:21:00 y 0:21:10)

en una etapa posterior. Esta etapa en muchos casos incrementa considerablemente el número de grupos llegando hasta 96 grupos en el registro 207. Por ello, allí se aplica una última fase offline de fusión de grupos, limitando a un máximo de 25 el número de grupos por registro. También indicar que en [20] se ha parametrizado y adaptado el método utilizando conocimiento experto de cardiólogos y valores extraídos de la bibliografía de cardiología, algo que se ha evitado en este método y que podría haber mejorado el rendimiento.

Casi la totalidad de latidos del tipo E, que representa a los latidos ventriculares de escape, aparecen en el registro 207 en el que es posible apreciar que, aun siendo del mismo tipo, los latidos pueden variar considerablemente en morfología e incluso en la distancia entre latidos (véase Figura 5.4). Por ello, este tipo de latidos resultan extremadamente difíciles para un clasificador [123]. Sin embargo, un algoritmo de agrupamiento, como el que aquí se presenta, puede separar fácilmente en distintos grupos las diversas morfologías de este tipo de latidos (véase Tabla 5.4).

Nótese que EPN-EAC puede asumir adecuadamente un incremento del número de características que representan al latido. Esto permite incluir muchas otras características como la potencia espectral, la media, la curtosis, etc. Estas características podrían agruparse en los vectores que fueran necesarios y que posteriormente servirían para generar particiones mediante K-means. La complejidad del proceso de extracción de evidencia depende únicamente del parámetro o elegido y del número de particiones generadas, pero no del

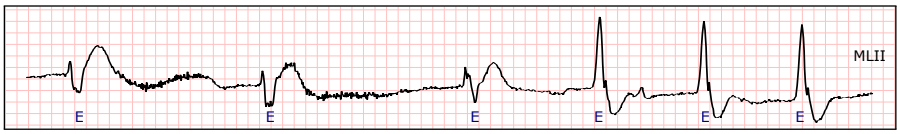


Figura 5.4: Fragmento de un ECG con latidos del tipo E, con distintas morfologías y distancia entre latidos. (Fuente: MIT-BIH Arrhythmia Database, registro 207, entre 0:27:20 y 0:27:30)

número de características que representan el latido. Por otra parte, a la hora de buscar la pareja más similar la complejidad del tercer criterio utilizado (5.7), en el que se calcula la distancia euclídea, sí que depende del número de características utilizadas para representar al latido, pero generalmente no será aplicado (en las pruebas realizadas se aplicó en menos de un 1 % de las ocasiones). También indicar que en el algoritmo por defecto se extrae evidencia con cada objeto nuevo que se recibe, pero esta acción podría ejecutarse solo cuando un número determinado de nuevos objetos estén disponibles para mejorar la eficiencia computacional.

En el caso de utilizar las anotaciones de la AAMI (véase Tabla 5.5), es posible comparar nuestro trabajo con [20], [30], [31], [77] y [85]. También se comparan los resultados obtenidos con los de PN-EAC del capítulo anterior. Podemos ver esta comparación en la Tabla 5.6, donde se aprecia que EPN-EAC y PN-EAC obtienen unos resultados comparables al resto de resultados en la bibliografía y mejores en varios casos. EPN-EAC obtiene mayor sensibilidad para latidos ventriculares (V), mayor valor predictivo positivo para los latidos de fusión (F), mayor F1 para latidos normales (N) y mayor precisión total. PN-EAC por otra parte obtiene mayor sensibilidad para latidos normales, mayor valor predictivo positivo para latidos supraventriculares (S), latidos ventriculares y latidos inclasificables (Q), junto con mayor F1 para latidos ventriculares, de fusión e inclasificables. Por tanto, entre ambos algoritmos obtienen los mejores resultados para casi todos los tipos de latido.

El valor F1 nos da una medida general del rendimiento del algoritmo en una determinada clase, combinando sensibilidad y valor predictivo positivo. El algoritmo EPN-EAC obtiene el mejor resultado para los latidos normales, que son el grupo más numeroso en la base de datos. Por ello, la precisión total será mayor para este algoritmo. Además, para los latidos ventriculares e inclasificables solo es superado en este valor por el algoritmo PN-EAC. Estas tres clases de latidos juntas representan el 96.65 % de los latidos de la base de datos. Los peores resultados se obtienen para los latidos supraventriculares (S), que requieren de información de la onda P o información de la distancia entre latidos.

Por otra parte, el algoritmo PN-EAC obtiene el mejor F1 para los latidos ventriculares, de fusión e inclasificables, pero al ser estos tipos de latidos menos numerosos en la base de datos la precisión total es algo más baja, aunque cercana a la obtenida por el resto de métodos de la bibliografía.

Tabla 5.6: Comparación con trabajos previos utilizando las anotaciones recomendadas por la AAMI. "Se" representa la sensibilidad, "P+" el valor predictivo positivo y "F1" el Valor-F. † son los resultados obtenido por EPN-EAC en este capítulo y ‡ los resultados obtenidos por PN-EAC en el capítulo anterior.

	N			S			V			F			Q			
	Se	P+	F1	Se	P+	F1	Se	P+	F1	Se	P+	F1	Se	P+	F1	Acc
†	99.66	99.25	**99.45**	84.09	90.59	87.22	**98.55**	97.06	97.80	64.88	**94.04**	76.78	99.61	99.95	99.78	**98.89**
‡	**99.84**	98.77	99.31	71.00	**95.38**	81.40	97.09	**98.81**	**97.95**	81.20	86.13	**83.59**	99.68	**99.96**	**99.82**	98.71
Castro2015	99.58	99.25	99.41	87.54	92.01	**89.72**	96.34	96.49	96.41	75.56	87.57	81.12	99.32	99.7	99.51	98.84
Lagerholm2000	99.05	**99.66**	99.35	**90.89**	83.1	86.82	98.01	96.05	97.02	87.32	75.06	80.73	**99.95**	99.56	99.75	98.77
DeChazal2004	86.86	99.16	92.60	75.94	38.53	51.12	77.74	81.59	79.62	89.43	8.57	15.64	0	0	-	85.88
DeChazal2006	94.3	99.36	96.76	87.72	46.95	61.16	94.34	94.3	94.32	73.97	29.15	41.82	0	0	-	93.89
Llamedo2011	77.55	99.47	87.15	72.88	41.34	52.76	91.25	95.94	93.54	**94.69**	3.67	7.07	-	-	-	78.3

Conclusiones y trabajo futuro

En esta memoria hemos presentado un conjunto de soluciones que tienen por objeto el procesamiento eficiente del electrocardiograma con el fin último de su interpretación, y que abordan desde su representación hasta el agrupamiento morfológico del complejo QRS. En este capítulo se analizarán las principales aportaciones del trabajo realizado y su continuación en el futuro próximo.

A nivel de representación se ha optado por utilizar una base de funciones: la de polinomios de Hermite. Esta base de funciones constituye un conjunto ortonormal que proporciona un modelo paramétrico del complejo QRS. Si bien esta propuesta no es nueva en el ámbito del procesamiento electrocardiográfico, su uso adolece en la bibliografía científica de una cierta arbitrariedad en la selección del conjunto de polinomios a utilizar, cuya decisión carece de una argumentación objetiva, y se basa por regla general en términos meramente visuales. En la presente memoria se muestra un estudio de la representación óptima del complejo QRS mediante polinomios de Hermite, a partir de criterios basados en la teoría de la información, y utilizando como referencia medidas como AIC o BIC. Dicho estudio se ha aplicado a la *MIT-BIH Arrhythmia Database*, base de datos de referencia en el análisis computacional del electrocardiograma, y que presenta la mayor complejidad morfológica de latidos que se puede encontrar en una base de datos de referencia. El estudio aquí propuesto muestra una excesiva simplificación en la representación del complejo QRS que se realiza habitualmente en la bibliografía. En tanto que AIC y BIC son robustos al ruido blanco, esa simplificación excluye necesariamente de la representación a algunos de los fenómenos fisiológicos que subyacen en el trazado electrocardiográfico, quedando fuera de cualquier análisis posterior.

Así todo, el coste computacional del cálculo de la representación del complejo QRS basada en funciones de Hermite crece de un modo no lineal con el número de funciones. Por

consiguiente, una representación óptima del latido no es factible en la monitorización en tiempo real del electrocardiograma, o en aplicaciones de compresión eficaz de la señal. Se ha optado por tanto por diseñar un cálculo eficiente que explota el paralelismo inherente a las arquitecturas de GPU, cada vez más presentes en los sistemas de cómputo más accesibles. Para ello se ha seleccionado una arquitectura, CUDA, cuyo uso se haya ampliamente extendido. En la presente memoria se muestra una propuesta de los algoritmos de representación para CUDA, tanto para su ejecución en diferido como en tiempo real. En el procesamiento en diferido de registros de larga duración se consiguen alcanzar aceleraciones de hasta 186×; en el procesamiento en tiempo real se obtienen aceleraciones de hasta 110× cuando los latidos se procesan en grupos de 100, y aceleraciones de hasta 23× cuando se procesan en grupos de 5. Estas prestaciones permiten una representación en tiempo real del latido incluso en el caso de utilizar 30 funciones.

En la presente tesis se aborda el agrupamiento morfológico del latido cardiaco como la herramienta más eficaz para el análisis de los trastornos de conducción del miocardio. En este sentido, se presenta una nueva técnica de agrupamiento estático, basada en la acumulación de evidencia, denominada PN-EAC, (*Positive and Negative Evidence Accumulation*). Esta técnica pertenece al paradigma de agrupamiento mediante *ensembles*, consistente en la generación y posterior combinación de múltiples particiones obtenidas por distintos algoritmos de agrupamiento, o por un mismo algoritmo de agrupamiento empleando distintos parámetros o distintas representaciones de los datos. A través de la combinación de múltiples resultados de agrupamiento es posible mejorar la robustez y estabilidad de los resultados individuales, superando las limitaciones expresivas de cada uno de los agrupamientos particulares que constituyen el *ensemble*.

Dentro de las técnicas de agrupamiento mediante *ensembles*, PN-EAC se enmarca en los algoritmos basados en la acumulación de evidencia. En este paradigma la agrupación de dos instancias en un mismo grupo de una de las particiones del *ensemble* se considera como un voto a favor de que dichas instancias se incluyan en el mismo grupo de la partición final. La idea subyacente en este paradigma es que aquellas instancias que en la partición natural de los datos se agrupan juntas tenderán a aparecer juntas en las distintas particiones del *ensemble*. Las coocurrencias de las instancias en los grupos de las particiones individuales se registran en una matriz de evidencia, matriz que puede considerarse como una matriz de similitud de los datos. Para obtener la partición final de datos a partir de dicha matriz de evidencia es posible emplear, por ejemplo, un algoritmo de agrupamiento jerárquico.

La técnica PN-EAC introduce un concepto novedoso en el paradigma de la agrupación mediante acumulación de evidencia: el de evidencia negativa. En la técnica PN-EAC además de extraerse evidencia positiva, se extrae también evidencia negativa de un segundo *ensemble* de particiones. Este segundo *ensemble* debe generarse a partir de una representación diferente de las instancias, de modo que la similitud en las características contempladas en dicha representación proporcione una evidencia débil de que las instancias deben agruparse juntas, pero la existencia de diferencias en dichas características proporciona evidencia de que dichas instancias no deben agruparse juntas (evidencia negativa).

La técnica PN-EAC genera una matriz de evidencia combinando evidencia positiva y evidencia negativa, y a partir de dicha matriz obtiene la partición final. En su aplicación al agrupamiento de latidos, se ha utilizado la base de datos *MIT-BIH Arrhythmia Database* para la validación y comparación de PN-EAC con otras técnicas de agrupamiento previas, utilizando las anotaciones de la base de datos para determinar el error en el agrupamiento, determinado por la reunión en el mismo grupo de latidos con anotaciones diferentes. Para ello, la evidencia positiva se extrae de la información derivada de la morfología del QRS y la negativa de la información derivada de la distancia entre latidos: el hecho de que dos latidos presenten una distancia claramente diferente con el latido previo y el siguiente proporciona una fuerte evidencia de que ambos latidos pertenecen a tipos diferentes y, por tanto, no deberían agruparse juntos. Los resultados obtenidos en el proceso de validación avalan la utilidad de la evidencia negativa para mejorar el resultado global del agrupamiento.

Una de las características más destacables de PN-EAC es su capacidad para explotar de manera eficiente la redundancia presente en los datos. Esta redundancia permite representar los datos de partida como un conjunto de múltiples fuentes de evidencia que se incorporan a la generación de particiones. Su aplicación en el procesamiento del electrocardiograma resulta evidente: la disponibilidad de hasta 12 derivaciones plantea la oportunidad de extraer evidencia de cada una de las 12 representaciones de un mismo latido, e integrar esa información sin incrementar apreciablemente ni la complejidad de la representación ni el coste de cómputo, al no integrar las múltiples derivaciones en un solo espacio de características. La memoria muestra unos resultados particularmente satisfactorios sobre la base de datos *INCART Database*, donde se demuestra que la incorporación progresiva de más derivaciones en el agrupamiento mejora su resultado significativamente.

Como evolución de PN-EAC se presenta una técnica de agrupamiento dinámico, denominada EPN-EAC, (*Evolving Positive and Negative Evidence Accumulation*), que busca

adaptar el paradigma de acumulación de evidencia a aquellos escenarios de monitorización continua, en los que no se dispone de todo el registro electrocardiográfico al inicio del agrupamiento, y se desea que los resultados del agrupamiento evolucionen a lo largo de la monitorización. La solución desarrollada emplea una lista de tamaño constante para almacenar un resumen representativo de las distintas formas de latidos observados hasta un momento dado. Cada vez que un nuevo latido está disponible se busca dentro de la lista el par de latidos más similares, que se fusionan calculando el promedio de sus vectores de características, dejando así un espacio libre para el nuevo latido. A continuación, se extrae nueva evidencia de los latidos presentes; dicha evidencia se añade a la recopilada hasta ese momento. Finalmente, si se desea, se obtiene una partición final sobre los datos disponibles hasta ese momento.

EPN-EAC tiene una complejidad computacional, tanto en uso de CPU como de memoria, cuadrático respecto al tamaño de la lista, reduciendo el coste computacional que muestra el algoritmo PN-EAC, cuadrático respecto al tamaño total del registro. Y sin embargo, la validación del nuevo agrupamiento respecto a la *MIT-BIH Arrhythmia Database* muestra resultados incluso mejores, que se explican por el mayor número de particiones que se realizan en el transcurso del agrupamiento dinámico con las sucesivas actualizaciones de la lista.

Tanto PN-EAC como EPN-EAC se han sometido a procesos de validación con bases de datos públicas que permiten su comparación con otras técnicas de la bibliografía científica. Dicha validación muestra unos resultados comparables e incluso superiores a los de las mejores técnicas publicadas hasta la fecha en lo que respecta a algunos de los términos de la comparación. Pero no se puede decir con rotundidad que una sola de las técnicas destaque sobre las demás. En este sentido, cabe valorar que tanto PN-EAC como EPN-EAC, si bien muestran un excelente comportamiento en el dominio electrocardiográfico, se presentan como técnicas generales de propósito general, y por tanto aplicables a conjuntos generales de problemas. En ese sentido se apartan de algunas de las mejores técnicas de agrupamiento morfológico de latidos, diseñadas específicamente para un buen comportamiento en este problema, pero difícilmente extrapolable a otros.

Respecto al trabajo futuro, varias son las líneas de continuación del trabajo de investigación aquí iniciado:

– El carácter general que se ha deseado para el funcionamiento de las técnicas PN-EAC y EPN-EAC ha excluido la utilización de criterios específicos basados en el

conocimiento del dominio. Sin embargo, se reconoce en el ámbito del aprendizaje automático un interés en incluir dicho conocimiento en forma de algoritmos de aprendizaje semisupervisado. En particular, en el problema del agrupamiento morfológico de latidos sería interesante disponer, por ejemplo, de una caracterización de artefactos de señal que evite la multiplicación de grupos en señales ruidosas. Además, dicha caracterización se puede extender a ciertas morfologías bien conocidas e identificadas con el origen de activación del complejo QRS, lo que proporcionaría utilidades de clasificación al médico especialista.

– Si llevamos un paso más allá la estrategia de semisupervisión, es posible desarrollar a partir de las técnicas PN-EAC y EPN-EAC un clasificador asistido, en el que el médico intervenga en el etiquetado de algunos latidos de cada registro electrocardiográfico, para posteriormente asignarse etiquetas de manera automática al resto de los latidos del registro a partir de las operaciones de agrupamiento.

– Tanto PN-EAC como EPN-EAC han mostrado un mejor comportamiento con un número prefijado de grupos. Sin embargo, creemos que es posible diseñar nuevos mecanismos de control del número óptimo de grupos que eviten su proliferación innecesaria. Entendemos que el uso de medidas de la teoría de la información puede proporcionar indicadores robustos para ello.

USC
UNIVERSIDADE
DE SANTIAGO
DE COMPOSTELA

Conclusions and future work

This thesis presents a set of solutions aimed at efficiently processing ECG for subsequent interpretation purposes. These solutions range from the representation to the morphological clustering of the QRS complex. In this chapter the main achievements made and the forthcoming steps are analysed in detail.

Hermite basis functions have been adopted for the representation of the QRS complex. These basis functions are orthonormal and provide a useful parametric model. Although this proposal is not new in the scientific bibliography, there is a lack of rationale for the number of functions to be used for QRS representation, and its choice is usually based on visual assessment. This thesis presents a rigorous study on the optimal representation of the QRS complex by means of Hermite basis functions, through the use of criteria founded on information theory, the Akaike Information Criterion (AIC) and the Bayesian Information Criterion (BIC). This study has been applied to the MIT-BIH Arrhythmia Database, the gold standard for computerised ECG analysis due to its morphological variability. The number of Hermite functions varied from 2 to 30, and the impact of using different filtering strategies (only low pass, only high pass, low pass and high pass, and no filtering) were also tested. Our study highlights the usual oversimplification of the representation in the bibliography and, given that AIC and BIC have proven to be robust against white noise, this oversimplification excludes from the representation some physiological processes underlying the electrocardiographical trace, preventing their analysis and interpretation.

The cost of computing the Hermite function representation shows a nonlinear growth with the number of functions. Our CPU-based implementation is capable of processing ECG recordings in real-time using up to 16 functions. Therefore, an optimal representation of each beat is not feasible in real time monitoring or in data compressing applications. An efficient calculation is proposed by exploiting GPU-based parallel computing, a

cost-effective and high-performance alternative to traditional CPUs. The CUDA platform has been chosen for the implementation, due to its good stability, its ease of use, and its wide adoption by the scientific community. This thesis presents CUDA implementations, both for offline and for online real time purposes. In offline processing, experiments with long term ECG recordings yield a speedup of about 186×; in online processing, experiments yield a speedup of about 110× when beats are processed in groups of size 100, and a speedup of about 23× when beats are processed in groups of size 5. This performance enables real time representation of the QRS complex, even when using 30 Hermite functions.

The present work is based on the assumption that beat clustering is the most effective tool for the analysis of conduction disorders. In this sense, a new method for static clustering is provided, PN-EAC (Positive and Negative Evidence Accumulation), based on the paradigm of ensemble clustering, also known as consensus clustering. This paradigm proposes to combine multiple partitions of a given data set in order to obtain a final partition that is better than the individual ones. The individual partitions can be obtained either by applying the same clustering algorithm with different parameters, by different clustering algorithms, or by using different data representations. By combining multiple clustering results, ensemble clustering can improve robustness, stability and accuracy of each individual result, and it can generate partitions that do not have to follow rigid models, such as hyperspherical or hyperelipsoidal partitions, but partitions that may have arbitrary shapes.

As a method for ensemble clustering, PN-EAC is based on the accumulation of evidence. The idea of evidence accumulation clustering is to combine the results of multiple clusterings into a single data partition by viewing each clustering result as independent evidence of data organization. Thus, beats belonging to a natural cluster are more likely to be placed in the same cluster in the different partitions. A voting mechanism to combine the clustering results is proposed, assuming that the co-occurrence of a pair of beats in the same cluster sums a vote to group together these beats in the final partition. The co-occurrences of pairs of beats are mapped into a co-association matrix, that can be considered as a similarity matrix. In order to obtain the final partition, a different clustering algorithm can be applied over this new similarity matrix (e.g., a hierarchical algorithm).

PN-EAC introduces a novel concept in evidence accumulation clustering: negative evidence. PN-EAC extracts both positive and negative evidence from the data set and combines it into an evidence matrix. A second ensemble copes with the representation of the beats used to extract negative evidence. Despite similarities between a couple of beats in this

representation provide no evidence to support their grouping in the same natural cluster, certain differences between a pair of beats provide evidence that they should not be grouped together in the final partition. The concept of negative evidence was designed to adequately exploit information extracted from the distances between beats, necessary for the identification of certain types of arrhythmias, such as premature beats. There is a wide variety of types of heartbeats that can occur with similar distances to the previous and next heartbeat. Therefore, the information derived from the distances between beats is a poor source of positive evidence to group together a pair of beats. However, the fact that two beats have markedly different values of the distance with the previous and next heartbeat provides strong evidence that these beats belong to different types and should not be grouped together.

The MIT-BIH Arrhythmia Database was used for validation and evaluation purposes. Three different strategies for the generation of evidence were tested: (1) combining all available features of the beat into a single feature vector and generating positive evidence using this representation; (2) creating an independent feature vector with the QRS morphology information for each available lead, and an additional vector with the information derived from the distance between beats, and generating positive evidence for all these representations; and (3) using the same feature vectors as in the previous case, but extracting negative evidence from the information derived from the distances between beats. By applying these strategies over the MIT-BIH database and setting the number of groups to 25, a percentage of incorrectly grouped beats of 2.25%, 1.81% and 1.44% were obtained for each of the three strategies. The second and third strategies use the same feature vectors to generate the partitions, being the only difference that from the partitions generated using the distances between beats negative evidence is extracted. Therefore, the improvement between both strategies ($p = 0.0404$) is exclusively due to the use of negative evidence. This result demonstrates the usefulness of this concept for the grouping of beats, and supports the general interest of the PN-EAC technique.

A remarkable feature of PN-EAC is its ability to exploit data redundancy. PN-EAC represents the data as a set of multiple sources of evidence that shape data partitions. Beat clustering can benefit from redundancy, since the ECG is typically recorded with a multiple lead configuration (up to 12 ECG leads) and all this information can be easily integrated with a computational cost linear in the number of sources of evidence. This hypothesis was validated over the 71 recordings with 12 leads of the INCARTDB Database. Tests were run using 2, 4, 6, 8, 10 and 12 leads from the database. The addition of leads always reduced the

average error on the database ($p < 0.001$, for each number of leads versus the immediately preceding). When only two leads were used and negative evidence was extracted from the information derived from the distances between beats, the average error on the database was 0.601%. This error was reduced to 0.338% when 12 leads were used.

The PN-EAC technique when applied to the electrocardiographic domain has the advantages of not imposing a fixed shape on the beat clusters, and of providing a simple solution for integrating information from an arbitrary number of leads. Furthermore, the algorithm is highly parallelizable, since the extraction of evidence from each of the leads can be performed independently. However, this technique has two major limitations. The first one is its computational cost, which both in terms of CPU time and memory requirements is quadratic in the total number of beats, making it difficult to use this technique for the analysis of long-term recordings (for example, Holter recordings). The second one is that this technique requires that all the beats to be grouped are available from the beginning and cannot therefore be applied for real-time monitoring of the patient.

As an evolution of PN-EAC, a new method of dynamic clustering is provided, EPN-EAC (Evolving Positive and Negative Evidence Accumulation), aimed to adapting the evidence accumulation paradigm to those scenarios of continuous monitoring, requiring that the results evolve over time. EPN-EAC uses a fixed size list to save a representative summary of the different types of beats recorded up to a particular point in time. Every new beat available triggers a search along this list of the two most similar beats, and they are combined in a new average beat, thus leaving room in the list for the new beat. Then, new evidence is obtained from the present summary beats, and this evidence is added to the available evidence. Finally, the final partition can be obtained for the ECG recording up to the present time.

Both the time and space complexity of EPN-EAC are $O(n^2)$, where n is the size list. A significant improvement is made with respect to PN-EAC, which is $O(n^2)$, being n the size of the whole recording. Furthermore, validation of EPN-EAC against MIT-BIH Arrhythmia Database yields even better results than PN-EAC: (1.41% vs. 1.44%, p =0.0039). This can be explained by a greater number of partitions computed in the dynamic algorithm. In this regard, we must remark that the calculation of an equivalent number of partitions in the static algorithm would be a challenge due to the greater computational complexity of PN-EAC. Therefore, EPN-EAC solves the two main limitations of PN-EAC by enabling real-time processing of long-term recordings with an affordable computational cost, while maintaining the advantages of the static version: it does not impose any rigid form on the clusters it

creates, and it provides a simple solution for integrating information from multiple ECG leads.

Both PN-EAC and EPN-EAC have been validated against public databases, in order to compare them with previous proposals in the scientific bibliography (MIT-BIH Arrhythmia Database and INCARTDB Database). The validation results are comparable or even better to previous results in some terms, but none of them can be clearly declared the winner. In this sense, both PN-EAC and EPN-EAC, beyond their excellent performance in the electrocardiographic domain, were designed as domain-independent solutions to clustering problems, following a different strategy than the best previous beat clustering methods, which were specifically designed for the electrocardiographic domain and are hardly applicable to other domains.

A number of issues should be addressed in the future work. The main research directions to explore can be summarized as follows:

– In order to provide new domain-independent methods, PN-EAC and EPN-EAC do not involve specific criteria from electrocardiographic knowledge. However, the machine learning community judges as a promising path the incorporation of well-established knowledge in new semisupervised methods. In this regard, it could be of interest to characterize typical signal artefacts for limiting cluster proliferation in noisy recordings. Furthermore, this characterization can be extended to certain well-known morphologies according to the beat origin, simplifying the interpretation task.

– Taking a further step in a semisupervised strategy, it is possible to develop from PN-EAC and EPN-EAC a classification method by requiring from the cardiologist the labelling of some beats of a given recording. Then, the clustering could assign labels to the remaining beats of the recording.

– Both PN-EAC and EPN-EAC have shown a better performance with a fixed number of groups. Nevertheless, further efforts should be made to design new methods for delivering the optimal number of clusters. Information theory can provide robust measures to this end.

USC
UNIVERSIDADE
DE SANTIAGO
DE COMPOSTELA

Bibliografía

[1] R. Acharya, S. M. Krishnan, J. A. Spaan, and J. S. Suri. *Advances in cardiac signal processing*. Springer, 2007.

[2] C. C. Aggarwal. *Data streams: models and algorithms*, volume 31. Springer Science & Business Media, 2007.

[3] N. Ahmed, P. Milne, and S. Harris. Electrocardiographic data compression via orthogonal transforms. *IEEE Trans. Biomed. Eng.*, (6):484–487, 1975.

[4] T. Ajam and A. A. Mehdirad. Electrocardiography, 2016.

[5] H. Akaike. A new look at the statistical model identification. *Automatic Control, IEEE Transactions on*, 19(6):716–723, 1974.

[6] M. M. Al Rahhal, Y. Bazi, N. Alajlan, S. Malek, H. Al-Hichri, F. Melgani, and M. A. Al Zuair. Classification of aami heartbeat classes with an interactive elm ensemble learning approach. *Biomedical Signal Processing and Control*, 19:56–67, 2015.

[7] E. Alickovic and A. Subasi. Medical decision support system for diagnosis of heart arrhythmia using dwt and random forests classifier. *Journal of medical systems*, 40(4):1–12, 2016.

[8] R. V. Andreão, B. Dorizzi, and J. Boudy. ECG signal analysis through hidden Markov models. *IEEE Transactions on Biomedical Engineering*, 53(8):1541–9, Aug. 2006.

[9] B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom. Models and issues in data stream systems. In *Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 1–16. ACM, 2002.

[10] A. L. Bakker, G. Nijkerk, B. E. Groenemeijer, R. A. Waalewijn, E. M. Koomen, R. L. Braam, and H. J. Wellens. The lewis lead making recognition of p waves easy during wide QRS complex tachycardia. *Circulation*, 119(24):e592–e593, 2009.

[11] B. Bakker and T. Heskes. Clustering ensembles of neural network models. *Neural networks*, 16(2):261–269, 2003.

[12] R. Bellman. *Dynamic Programming (Dover Books on Computer Science)*. Dover Publications, 2013.

[13] R. E. Bellman. *Adaptive control processes: a guided tour*. Princeton university press, 2015.

[14] M. Blanco-Velasco, B. Weng, and K. E. Barner. ECG signal denoising and baseline wander correction based on the empirical mode decomposition. *Computers in biology and medicine*, 38(1):1–13, Jan. 2008.

[15] A. Bouchachia. Evolving clustering: An asset for evolving systems. *IEEE SMC Newsl*, 36, 2011.

[16] G. Braccini and L. Edenbrandt. Self-organizing maps and Hermite functions for classification of ECG complexes. *in Cardiology 1997*, 24:425–428, 1997.

[17] K. P. Burnham and D. R. Anderson. *Model selection and multimodel inference: a practical information-theoretic approach*. Springer Science & Business Media, 2003.

[18] K. P. Burnham and D. R. Anderson. Multimodel inference understanding aic and bic in model selection. *Sociological methods & research*, 33(2):261–304, 2004.

[19] CARDIOESPORT. Imagen corazón.

[20] D. Castro, P. Félix, and J. Presedo. A method for context-based adaptive QRS clustering in real time. *IEEE journal of biomedical and health informatics*, 19(5):1660–1671, 2015.

[21] V. Chudacek, M. Petrík, G. Georgoulas, M. Cepek, L. Lhotská, and C. Stylios. Comparison of seven approaches for holter ECG clustering and classification. In *Engineering in Medicine and Biology Society, 2007. EMBS 2007. 29th Annual International Conference of the IEEE*, pages 3844–3847. IEEE, 2007.

[22] G. Claeskens, N. L. Hjort, et al. *Model selection and model averaging*, volume 330. Cambridge University Press Cambridge, 2008.

[23] G. Clifford, F. Azuaje, and P. McSharry. *Advanced methods and tools for ECG data analysis*. Artech House, 2006.

[24] C. S. I. Company. QRS complex diagram, 1950.

[25] E. Cosson, F. Paycha, J. Paries, S. Cattan, a. Ramadan, D. Meddah, J.-R. Attali, and P. Valensi. Detecting silent coronary stenoses and stratifying cardiac risk in patients with diabetes: ECG stress test or exercise myocardial scintigraphy? *Diabetic medicine : a journal of the British Diabetic Association*, 21(4):342–8, Apr. 2004.

[26] D. Cuesta-Frau, M. O. Biagetti, R. A. Quinteiro, P. Mico-Tormos, and M. Aboy. Unsupervised classification of ventricular extrasystoles using bounded clustering algorithms and morphology matching. *Medical and Biological Engineering and Computing*, 45(3):229–239, 2007.

[27] D. Cuesta-Frau, J. C. Pérez-Cortés, and G. Andreu-García. Clustering of electrocardiograph signals in computer-aided holter analysis. *Computer methods and programs in Biomedicine*, 72(3):179–196, 2003.

[28] W. H. Day and H. Edelsbrunner. Efficient algorithms for agglomerative hierarchical clustering methods. *Journal of classification*, 1(1):7–24, 1984.

[29] P. De Chazal. An adapting system for heartbeat classification minimising user input. In *Engineering in Medicine and Biology Society (EMBC), 2014 36th Annual International Conference of the IEEE*, pages 82–85. IEEE, 2014.

[30] P. de Chazal, M. O'Dwyer, and R. B. Reilly. Automatic classification of heartbeats using ECG morphology and heartbeat interval features. *IEEE Transactions on Biomedical Engineering*, 51(7):1196–206, July 2004.

[31] P. de Chazal and R. B. Reilly. A patient-adapting heartbeat classifier using ECG morphology and heartbeat interval features. *IEEE Transactions on Biomedical Engineering*, 53(12 Pt 1):2535–43, Dec. 2006.

[32] W. R. de Holanda-Miranda, F. M. Furtado, P. M. Luciano, and A. Pazin-Filho. Lewis lead enhances atrial activity detection in wide QRS tachycardia. *The Journal of emergency medicine*, 43(2):e97–e99, 2012.

[33] G. de Lannoy, D. Francois, J. Delbeke, and M. Verleysen. Weighted conditional random fields for supervised interpatient heartbeat classification. *Biomedical Engineering, IEEE Transactions on*, 59(1):241–247, Jan 2012.

[34] L. Y. Di Marco and L. Chiari. A wavelet-based ECG delineation algorithm for 32-bit integer online processing. *Biomedical engineering online*, 10(1):23, 2011.

[35] T. G. Dietterich. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine learning*, 40(2):139–157, 2000.

[36] S. Dudoit and J. Fridlyand. Bagging to improve the accuracy of a clustering procedure. *Bioinformatics*, 19(9):1090–1099, 2003.

[37] A.-A. EC57. Testing and reporting performance results of cardiac rhythm and st segment measurement algorithms. *Association for the Advancement of Medical Instrumentation, Arlington, VA*, 1998.

[38] L. Edenbrandt and O. Pahlm. Vectorcardiogram synthesized from a 12-lead ECG: superiority of the inverse dower matrix. *Journal of electrocardiology*, 21(4):361–367, 1988.

[39] V. Estivill-Castro. Why so many clustering algorithms: a position paper. *ACM SIGKDD explorations newsletter*, 4(1):65–75, 2002.

[40] M. Fernández-Delgado and S. Barro Ameneiro. MART: a multichannel ART-based neural network. *IEEE Transactions on Neural Networks*, 9(1):139–50, Jan. 1998.

[41] H. Finner. On a monotonicity problem in step-down multiple test procedures. *Journal of the American Statistical Association*, 88(423):920–923, 1993.

[42] A. Fred. Finding consistent clusters in data partitions. In *Multiple classifier systems*, pages 309–318. Springer, 2001.

[43] A. L. Fred and A. K. Jain. Combining multiple clusterings using evidence accumulation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(6):835–850, 2005.

[44] J. Friedman, T. Hastie, and R. Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics Springer, Berlin, 2001.

[45] J. H. Friedman. On bias, variance, 0/1—loss, and the curse-of-dimensionality. *Data mining and knowledge discovery*, 1(1):55–77, 1997.

[46] M. Friedman. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the american statistical association*, 32(200):675–701, 1937.

[47] M. M. Gaber, A. Zaslavsky, and S. Krishnaswamy. Mining data streams: a review. *ACM Sigmod Record*, 34(2):18–26, 2005.

[48] L. Galvani and G. Aldini. *De Viribus Electricitatis In Motu Musculari Comentarius Cum Joannis Aldini Dissertatione Et Notis; Accesserunt Epistolae ad animalis electricitatis theoriam pertinentes*. Apud Societatem Typographicam, 1792.

[49] V. M. García-Molla, A. Liberos, A. Vidal, M. Guillem, J. Millet, A. Gonzalez, F.-J. Martínez-Zaldívar, and A. M. Climent. Adaptive step ode algorithms for the 3d simulation of electric heart activity with graphics processing units. *Computers in biology and medicine*, 44:15–26, 2014.

[50] L. Gaztañaga, F. E. Marchlinski, and B. P. Betensky. Mecanismos de las arritmias cardiacas. *Revista Española de Cardiología*, 65(2):174–185, 2012.

[51] R. Ghaemi, M. N. Sulaiman, H. Ibrahim, N. Mustapha, et al. A survey: clustering ensembles techniques. *World Academy of Science, Engineering and Technology*, 50:636–645, 2009.

[52] S. Ghosh and S. K. Dubey. Comparative analysis of k-means and fuzzy c-means algorithms. *International Journal of Advanced Computer Science and Applications*, 4(4):35–39, 2013.

[53] M. Goldman et al. Principles of clinical electrocardiography. 1976.

[54] Y. Goletsis, C. Papaloukas, D. I. Fotiadis, A. Likas, and L. K. Michalis. Automated ischemic beat classification using genetic algorithms and multicriteria decision analysis. *IEEE Transactions on Biomedical Engineering*, 51(10):1717–25, Oct. 2004.

[55] M. Hadjem, O. Salem, and F. Naït-Abdesselam. An ECG monitoring system for prediction of cardiac anomalies using wban. In *e-Health Networking, Applications and Services (Healthcom), 2014 IEEE 16th International Conference on*, pages 441–446. IEEE, 2014.

[56] P. S. Hamilton and W. J. Tompkins. Quantitative investigation of QRS detection rules using the mit/bih arrhythmia database. *IEEE transactions on biomedical engineering*, (12):1157–1165, 1986.

[57] H. Haraldsson, L. Edenbrandt, and M. Ohlsson. Detecting acute myocardial infarction in the 12-lead ECG using Hermite expansions and neural networks. *Artificial intelligence in medicine*, 32(2):127–36, Oct. 2004.

[58] T. K. Ho, J. J. Hull, and S. N. Srihari. Decision combination in multiple classifier systems. *IEEE transactions on pattern analysis and machine intelligence*, 16(1):66–75, 1994.

[59] S. Holm. A simple sequentially rejective multiple test procedure. *Scandinavian journal of statistics*, pages 65–70, 1979.

[60] M. R. Homaeinezhad, M. Erfanianmoshiri-Nejad, and H. Naseri. A correlation analysis-based detection and delineation of ECG characteristic events using template waveforms extracted by ensemble averaging of clustered heart cycles. *Comput. Biol. Med.*, 44:66–75, Jan. 2014.

[61] Y. Hong, S. Kwong, Y. Chang, and Q. Ren. Unsupervised feature selection using clustering ensembles and population based incremental learning algorithm. *Pattern Recognition*, 41(9):2742–2756, 2008.

[62] Y. Hu, W. Tompkins, and J. Urrusti. Applications of artificial neural networks for ECG signal detection and classification. *Journal of Electrocardiology*, 26:66–73, 1993.

[63] Y. H. Hu, S. Palreddy, and W. J. Tompkins. A patient-adaptable ECG beat classifier using a mixture of experts approach. *IEEE Transactions on Biomedical Engineering*, 44(9):891–900, Sept. 1997.

[64] R. a. Incalzi, L. Fuso, M. De Rosa, a. Di Napoli, S. Basso, G. Pagliari, and R. Pistelli. Electrocardiographic Signs of Chronic Cor Pulmonale : A Negative Prognostic Finding in Chronic Obstructive Pulmonary Disease. *Circulation*, 99(12):1600–1605, Mar. 1999.

[65] A. Jain. Data clustering: 50 years beyond K-means. *Pattern Recognition Letters*, 2010.

[66] R. Jane, S. Olmos, and P. Laguna. Adaptive Hermite models for ECG data compression: performance and evaluation with automatic wave detection. *Computers in Cardiology*, 1993.

[67] I. Jekova, G. Bortolan, and I. Christov. Assessment and comparison of different methods for heartbeat classification. *Medical Engineering & Physics*, 30(2):248–257, 2008.

[68] W. Jiang and S. Kong. Block-based neural networks for personalized ECG signal classification. *IEEE Trans. Neural Networks*, 18(6):1750–1761, Nov 2007.

[69] Kalumet. Principle of ekg/ECG formation, 2005.

[70] M. Karczewicz and M. Gabbouj. ECG data compression by spline approximation. *Signal Processing*, 59(1):43–59, 1997.

[71] A. Kashani and S. S. Barold. Significance of QRS complex duration in patients with heart failure. *Journal of the American College of Cardiology*, 46(12):2183–2192, 2005.

[72] S. Kiranyaz, T. Ince, and M. Gabbouj. Real-time patient-specific ECG classification by 1-d convolutional neural networks. *IEEE Transactions on Biomedical Engineering*, 63(3):664–675, 2016.

[73] S. Kiranyaz, T. Ince, J. Pulkkinen, and M. Gabbouj. Personalized long-term ECG classification: A systematic approach. *Expert Systems with Applications*, 38(4):3220–3226, Apr. 2011.

[74] D. B. Kirk and W. H. Wen-Mei. *Programming massively parallel processors: a hands-on approach*. Morgan Kaufmann, 2016.

[75] M. Korürek and A. Nizam. A new arrhythmia clustering technique based on ant colony optimization. *Journal of Biomedical Informatics*, 41(6):874–881, 2008.

[76] Y. Kutlu and D. Kuntalp. A multi-stage automatic arrhythmia recognition and classification system. *Computers in biology and medicine*, 41(1):37–45, 2011.

[77] M. Lagerholm, C. Peterson, G. Braccini, L. Edenbrandt, and L. Sörnmo. Clustering ECG complexes using hermite functions and self-organizing maps. *IEEE Transactions on Biomedical Engineering*, 47(7):838–48, July 2000.

[78] P. Laguna, R. Jané, and P. Caminal. Automatic detection of wave boundaries in multilead ECG signals: Validation with the CSE database. *Computers and biomedical research*, 27(1):45–60, 1994.

[79] P. Laguna, R. Jané, S. Olmos, N. V. Thakor, H. Rix, and P. Caminal. Adaptive estimation of QRS complex wave features of ECG signal by the hermite model. *Medical & Biological Engineering & Computing*, 34(1):58–68, Jan. 1996.

[80] L. Lam and S. Suen. Application of majority voting to pattern recognition: an analysis of its behavior and performance. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 27(5):553–568, 1997.

[81] T. Lewis, H. Feil, and W. Stroud. Observations upon flutter and fibrillation. ii. the nature of auricular flutter. *Heart*, 7(191245.6), 1920.

[82] P. Li, C. Liu, X. Wang, D. Zheng, Y. Li, and C. Liu. A low-complexity data-adaptive approach for premature ventricular contraction recognition. *Signal, Image and Video Processing*, 8(1):111–120, 2014.

[83] M. Llamedo, A. Khawaja, and J. Martínez. Analysis of 12-lead classification models for ECG classification. In *2010 Computing in Cardiology*, pages 673–676. IEEE, 2010.

[84] M. Llamedo, A. Khawaja, and J. P. Martinez. Cross-database evaluation of a multilead heartbeat classifier. *IEEE Transactions on Information Technology in Biomedicine*, 16(4):658–664, 2012.

[85] M. Llamedo and J. P. Martínez. Heartbeat classification using feature selection driven by database generalization criteria. *Biomedical Engineering, IEEE Transactions on*, 58(3):616–625, 2011.

[86] A. Lourenço, H. Silva, and A. Fred. Unveiling the biometric potential of finger-based ECG signals. *Computational intelligence and neuroscience*, 2011:5, 2011.

[87] E. J. d. S. Luz, W. R. Schwartz, G. Cámara-Chávez, and D. Menotti. ECG-based heartbeat classification for arrhythmia detection: A survey. *Computer methods and programs in biomedicine*, 127:144–164, 2016.

[88] Madhero88. Diagram showing the connection of ECG leads.

[89] S. Maheshwari, A. Acharyya, P. Rajalakshmi, P. E. Puddu, and M. Schiariti. Accurate and reliable 3-lead to 12-lead ECG reconstruction methodology for remote health monitoring applications. *IRBM*, 35(6):341–350, 2014.

[90] M. Malik, J. Bigger, A. Camm, R. Kleiger, A. Malliani, et al. Heart rate variability. *Circulation*, 93(5):1043–1065, 1996.

[91] N. Mantel. Chi-square tests with one degree of freedom; extensions of the mantel-haenszel procedure. *Journal of the American Statistical Association*, 58(303):690–700, 1963.

[92] J. B. Mark. *Atlas of cardiovascular monitoring*. Churchill Livingstone, 1998.

[93] R. J. Martis, C. Chakraborty, and A. K. Ray. A two-stage mechanism for registration and classification of ECG using gaussian mixture model. *Pattern Recognition*, 42(11):2979–2988, 2009.

[94] J. Mateo, A. Torres, A. Aparicio, and J. Santos. An efficient method for ECG beat classification and correction of ectopic beats. *Computers & Electrical Engineering*, 53:219–229, 2016.

[95] C. D. Mathers and D. Loncar. Projections of global mortality and burden of disease from 2002 to 2030. *PLoS medicine*, 3(11):e442, 2006.

[96] G. Moody and R. Mark. The impact of the MIT-BIH arrhythmia database. *Engineering in Medicine and Biology Magazine, IEEE*, 20(3):45 –50, may-june 2001.

[97] MoodyGroove. Various QRS complexes with nomenclature, 2007.

[98] G. Moruzzi. The electrophysiological work of carlo matteucci. *Brain research bulletin*, 40(2):69–91, 1996.

[99] F. Murtagh and P. Legendre. Ward's hierarchical agglomerative clustering method: which algorithms implement ward's criterion? *Journal of Classification*, 31(3):274–295, 2014.

[100] P. Nemenyi. Distribution-free multiple comparisons. In *Biometrics*, volume 18, page 263. INTERNATIONAL BIOMETRIC SOC 1441 I ST, NW, SUITE 700, WASHINGTON, DC 20005-2210, 1962.

[101] J. Nickolls, I. Buck, M. Garland, and K. Skadron. Scalable parallel programming with cuda. *Queue*, 6(2):40–53, 2008.

[102] J. Nickolls and W. J. Dally. The GPU computing era. *IEEE micro*, 30(2), 2010.

[103] S. P. I. of Cardiological Technics. *St. Petersburg Institute of Cardiological Technics 12-lead Arrhythmia Database*.

[104] H.-M. D. of Health Sciences and Technology. MIT-BIH arrhythmia database directory, 1997.

[105] S. Olmos, M. MillAn, J. Garcia, and P. Laguna. ECG data compression with the karhunen-loeve transform. In *Computers in Cardiology, 1996*, pages 253–256. IEEE, 1996.

[106] W. H. Organization et al. Who methods and data sources for country-level causes of death, 2000–2015, 2016.

[107] S. Osowski, L. T. Hoai, and T. Markiewicz. Support vector machine-based expert system for reliable heartbeat recognition. *IEEE Transactions on Biomedical Engineering*, 51(4):582–9, Apr. 2004.

[108] S. Osowski and M. Stodolski. On-line heart beat recognition using hermite polynomials and neuro-fuzzy network. *IEEE Transactions on Instrumentation and Measurement*, 52(4):1224–1231, Aug. 2003.

[109] J. Oster, J. Behar, O. Sayadi, S. Nemati, A. E. Johnson, and G. D. Clifford. Semisupervised ECG ventricular beat classification with novelty detection based on switching kalman filters. *IEEE Transactions on Biomedical Engineering*, 62(9):2125–2134, 2015.

[110] A. Otero and S. F. Dapena. A Low Cost Screening Test for Obstructive Sleep Apnea that can be Performed at the Patient's Home. *Communications*, pages 199–204, 2009.

[111] J. Pan and W. J. Tompkins. A real-time QRS detection algorithm. *IEEE transactions on biomedical engineering*, (3):230–236, 1985.

[112] K. Park, B. Cho, D. Lee, S. Song, J. Lee, Y. Chee, I. Kim, and S. Kim. Hierarchical support vector machine based heartbeat classification using higher order statistics and hermite basis function. In *2008 Computers in Cardiology*, pages 229–232. IEEE, Sept. 2008.

[113] K. Peason. On lines and planes of closest fit to systems of point in space. *Philosophical Magazine*, 2(11):559–572, 1901.

[114] H. V. Pipberger, E. D. Freis, L. Taback, and H. L. Mason. Preparation of electrocardiographic data for analysis by digital electronic computer. *Circulation*, 21(3):413–418, Mar. 1960.

[115] J. R. Quinlan. *C4. 5: programs for machine learning*, volume 1. Morgan kaufmann, 1993.

[116] A. Ramakrishnan and S. Saha. ECG coding by wavelet-based linear prediction. *IEEE Transactions on Biomedical Engineering*, 44(12):1253–1261, 1997.

[117] T. Reichlin, R. Abächerli, R. Twerenbold, M. Kühne, B. Schaer, C. Mueller, C. Sticherling, and S. Osswald. Advanced ECG in 2016: is there more than just a tracing? *Swiss medical weekly*, 146:w14303–w14303, 2016.

[118] M. Rivera-Ruiz, C. Cajavilca, and J. Varon. Einthoven's string galvanometer: the first electrocardiograph. *Texas Heart Institute Journal*, 35(2):174, 2008.

[119] I. Rodríguez-Fdez, A. Canosa, M. Mucientes, and A. Bugarín. STAC: a web platform for the comparison of algorithms using statistical tests. In *Proceedings of the 2015 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, 2015.

[120] J. L. Rodríguez-Sotelo, D. Peluffo-Ordoñez, D. Cuesta-Frau, and G. Castellanos-Domínguez. Unsupervised feature relevance analysis applied to improve ECG heartbeat clustering. *Computer Methods and Programs in Biomedicine*, 108(1):250 – 261, 2012.

[121] S. Saxena. Feature extraction from ECG signals using wavelet transforms for disease diagnostics. *International Journal of Systems Science*, 33(13):1073–1085, 2002.

[122] O. Sayadi, M. B. Shamsollahi, and G. D. Clifford. Robust detection of premature ventricular contractions using a wave-based bayesian framework. *IEEE Transactions on Biomedical Engineering*, 57(2):353–362, 2010.

[123] L. Schamroth and A. Dubb. Escape-capture bigeminy. mechanisms in sa block, av block, and reversed reciprocal rhythm. *British heart journal*, 27(5):667, 1965.

[124] G. Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464, 03 1978.

[125] H.-B. Shen and K.-C. Chou. Ensemble classifier for protein fold pattern recognition. *Bioinformatics*, 22(14):1717–1722, 2006.

[126] J. A. Silva, E. R. Faria, R. C. Barros, E. R. Hruschka, A. C. de Carvalho, and J. Gama. Data stream clustering: A survey. *ACM Computing Surveys (CSUR)*, 46(1):13, 2013.

[127] Ske. Les 12 dérivations d'un ECG, un cycle de chacune des 12 dérivations, sans quadrillage de fond., 2003.

[128] SKvalen. QRS complex diagram, 2007.

[129] R. R. Sokal. A statistical method for evaluating systematic relationships. *Univ Kans Sci Bull*, 38:1409–1438, 1958.

[130] L. Sörnmo, P. O. Börjesson, M. E. Nygå rds, and O. Pahlm. A method for evaluation of QRS shape features using a mathematical model for the ECG. *IEEE Transactions on Biomedical Engineering*, 28(10):713–7, Oct. 1981.

[131] L. Sörnmo and P. Laguna. *Bioelectrical signal processing in cardiac and neurological applications*. Elsevier Academic Press, 2005.

[132] O. G. Sotelo. *Manual de Arritmias Cardiacas: Guía Diagnóstica Terapéutica*. Editorial Universidad de Costa Rica, 2002.

[133] N. Sugiura. Further analysts of the data by akaike's information criterion and the finite corrections: Further analysts of the data by akaike's. *Communications in Statistics-Theory and Methods*, 7(1):13–26, 1978.

[134] Z. Syed, J. Guttag, and C. Stultz. Clustering and symbolic analysis of cardiovascular signals: discovery and visualization of medically relevant patterns in long-term data using limited prior knowledge. *EURASIP Journal on Applied Signal Processing*, 2007(1):97–97, 2007.

[135] N. V. Thakor, J. G. Webster, and W. J. Tompkins. Estimation of QRS complex power spectra for design of a QRS filter. *IEEE Transactions on biomedical engineering*, (11):702–706, 1984.

[136] N. V. Thakor and Y.-S. Zhu. Applications of adaptive filtering to ECG analysis: noise cancellation and arrhythmia detection. *IEEE transactions on biomedical engineering*, 38(8):785–794, 1991.

[137] A. Topchy, A. K. Jain, and W. Punch. Clustering ensembles: Models of consensus and weak partitions. *IEEE Transactions on pattern analysis and machine intelligence*, 27(12):1866–1881, 2005.

[138] A. P. Topchy, A. K. Jain, and W. F. Punch. A mixture model for clustering ensembles. In *SDM*, pages 379–390. SIAM, 2004.

[139] M. G. Tsipouras, D. I. Fotiadis, and D. Sideris. An arrhythmia classification system based on the RR-interval signal. *Artif. Intell. Med.*, 33(3):237–50, Mar. 2005.

[140] S. V. Vaseghi. *Advanced digital signal processing and noise reduction*. John Wiley & Sons, 2008.

[141] R. Von Borries, J. Pierluissi, and H. Nazeran. Wavelet transform-based ECG baseline drift removal for body surface potential mapping. In *Engineering in Medicine and Biology Society 2005.*, pages 3891–3894. IEEE, 2006.

[142] A. D. Waller. A demonstration on man of electromotive changes accompanying the heart's beat. *The Journal of physiology*, 8(5):229–234, 1887.

[143] Y. Wenyu, L. Gang, L. Ling, and Y. Qilian. ECG analysis based on pca and som. In *Neural Networks and Signal Processing, 2003. Proceedings of the 2003 International Conference on*, volume 1, pages 37–40. IEEE, 2003.

[144] D. J. Whellan, C. L. Green, J. P. Piccini, and M. W. Krucoff. QT as a safety biomarker in drug development. *Clinical pharmacology and therapeutics*, 86(1):101–4, July 2009.

[145] F. Wilcoxon. Individual comparisons by ranking methods. *Biometrics bulletin*, 1(6):80–83, 1945.

[146] R. Xu and D. Wunsch. Survey of clustering algorithms. *Neural Networks, IEEE Transactions on*, 16(3):645–78, May 2005.

[147] Q. Xue, Y. H. Hu, and W. J. Tompkins. Neural-network-based adaptive matched filtering for QRS detection. *IEEE Transactions on Biomedical Engineering*, 39(4):317–329, 1992.

[148] Y. Yang. Can the strengths of aic and bic be shared? a conflict between model indentification and regression estimation. *Biometrika*, 92(4):937–950, 2005.

[149] T. Y. Young and W. H. Huggins. On the representation of electrocardiograms. *IEEE Trans. Bio-Med. Electron.*, 10(3):86–95, 1963.

[150] D. Zhang. Wavelet approach for ECG baseline wander correction and noise reduction. In *Engineering in Medicine and Biology Society 2005.*, pages 1212–1215. IEEE, 2005.

[151] Q. Zhang, J. Wang, G. D. Guerrero, J. M. Cecilia, J. M. García, Y. Li, H. Pérez-Sánchez, and T. Hou. Accelerated conformational entropy calculations using graphic processing units. *Journal of chemical information and modeling*, 53(8):2057–2064, 2013.

[152] Z. Zhang, J. Dong, X. Luo, K.-S. Choi, and X. Wu. Heartbeat classification using disease-specific feature selection. *Comput. Biol. Med.*, 46:79–89, Mar. 2014.

Índice de figuras

USC
UNIVERSIDADE
DE SANTIAGO
DE COMPOSTELA

Índice de tablas