Implementación de un paquete de software para el ajuste de modelos FMM. Aplicación a la interpretación automática de la señal del electrocardiograma.

Adrián Lamela Pérez

Tutores:
Cristina Rueda Sabater
Carlos Vivaracho Pascual

30 de noviembre de 2020

Índice general

Abstract

Cyrcadian clock, cell cycle, astrophysics, are only the top of a much bigger iceberg of oscillatory signals. The study of this signals has been adressed since the decade of 90's, but it is now when the improvements of computer science allow us to achieve more results on these studies.

Many models have been developed, including Cosinor methodology and some machine learning techniques. However, the first one displays one major drawback, failing to represent a vast number of morphologies, while the second one acts as a "black box" with no enough precission.

This work focuses on a novel approach called *Frecuency Modulated Möbius* (FMM) developed by the research group "Inferencia con Restricciones" (University of Valladolid) in 2019. It also serves as the natural sequel to a previous work of this author [6]. Three papers establish the basis of the methodology and interesting applications. The first one describes the FMM methodology and was published in 2019 [1]. The second one studies the application of this model in an automatic analysis of electrocardiogram data, which is currently under revision [3]. The last one focuses on details of implementation, and it is still in development [2]. The author of this TFM has participated in the last two of them.

Chapter 1 describes the theoretical details of FMM model and its derivations, including the multicomponent FMM model, restricted FMM model, and ECG-based FMM model. In Chapter 2, we discuss in more depth the details and implications of an automatic analysis of ECGs based on this methodology. Last Chapter is focused on practical uses of these models, and examples of use of the software developed.

Introducción

El ciclo circadiano, el ciclo celular, el ciclo hormonal, la luz emitida por las estrellas, la actividad eléctrica del corazón o de las neuronas, procesos económicos... Todos ellos son datos de gran interés científico, expresados en forma de **señales oscilatorias**. Las características que definen un proceso oscilatorio son, a grandes rasgos, la amplitud, la fase, la forma, y el período. Generalmente, el período siempre se considera conocido, por lo que el interés radica en determinar las otras tres.

Para analizar este tipo de señales, se ha propuesto un gran número de modelos, desde los más sencillos y conocidos, como un modelo Cosinor o la descomposición de Fourier, hasta sofisticadas técnicas de aprendizaje automático que actúan como una caja negra de predicción. Los inconvenientes de los primeros radican básicamente en que no son lo suficientemente versátiles como para describir la morfología de situaciones complejas, como la electrocardiografía. En cuanto a los segundos, resulta difícil usarlos para convencer a expertos humanos de que son más fiables que ellos cuando no son capaces de decir claramente el porqué.

En este trabajo expondremos un modelo reciente, desarrollado y publicado en 2019, por parte del Grupo de Investigación Reconocido "Infenrencia con Restricciones" de la Universidad de Valladolid, en el que el autor de este trabajo ha participado desde 2018 [1]. También sirve como continuación del Trabajo de Fin de Grado presentado en 2019 por este mismo autor [6]. Este modelo, bautizado como *Frecuency Modulated Möbius*, FMM, ha demostrado una gran versatilidad para adaptarse a patrones simétricos y asimétricos, apuntados y no apuntados, que aparecen en la naturaleza en innumerables ocasiones.

Entre las principales ventajas de la utilización de este modelo destacamos dos. Por una parte, es un modelo paramétrico en el que surge una interpretación natural para las estimaciones realizadas, lo que solventa el problema de "caja negra" de las técnicas más sofisticadas. Además, con tan sólo dos parámetros más que uno de los modelos más sencillos, el Cosinor, abarca un espectro mucho más amplio de situaciones. Es por eso que el GIR se ha atrevido con uno de los retos más complicados y a la vez más interesantes de la última década: la interpretación automática de un ECG.

La actividad eléctrica del corazón se monitoriza con una máquina especial llamada electrocardiógrafo. Las señales producidas se proyectan en diferentes planos (más concretamente, 12), donde manualmente, un experto cardiólogo intuirá si existe un inminente infarto de miocardio, o si una arritmia puede ser o no peligrosa. No hace falta remarcar que la cantidad de experiencia necesaria para esta labor es increíblemente alta,

por lo que los recursos humanos no son especialmente abundantes.

Teniendo en mente esta situación, se expone un método de estimación e interpretación automático para un electrocardiograma, cuya implementación es el objetivo principal de este trabajo. Veremos cómo somos capaces de encontrar los componentes de la actividad eléctrica del corazón de forma sencilla para latidos típicos, y utilizando un algoritmo complejo de identificación para los más atípicos. La visión que presentamos aquí está aún en proceso de desarrollo, y nos limitaremos a señalar automáticamente las componentes de un ECG, pero no a tareas más complicadas como puede ser la detección de enfermedades. Ésta constituye una de las principales líneas de trabajo futuro. El contenido de esta parte del trabajo se apoya en otro artículo, que se encuentra actualmente en proceso de revisión [3], en la que el escritor de este trabajo figura como autor.

Las prometedoras perspectivas de la aplicacion del FMM para resolver problemas en muy diversos campos sugieren la necesidad de desarrollar un software específico para facilitar a los usuarios el acceso a la metodología. Ésta será la principal línea de acción de este trabajo. Para ello, se ha creado una librería de R. Esta librería, que pretende ser publicada en el repositorio oficial de R y estar disponible a finales de 2020, lleva a todos los investigadores la implementación del ajuste de una amplia variedad de modelos FMM. Se apoya en un artículo que está en sus momentos finales de redacción, y que será dirigido a la revista *Journal of Statistical Software*, en la que también participa el autor del trabajo que se presenta [2].

Las mayores dificultades que podemos encontrar en el análisis de este tipo de señales son la carencia de estructura y amplia carga computacional del modelo. En la aplicación del ECG, en particular, se necesita una capacidad computacional muy grande para llevar a cabo la estimación. El volumen de datos recogido por un electrocardiógrafo es muy elevado, y es necesario escoger los segmentos adecuados y aplicarles un preprocesamiento especial. Además, idealmente estaríamos hablando de una tarea de flujos de datos, en la que la captura precede a su inmediato análisis e interpretación automática. Hablaríamos de un sistema que capture un gran volumen de datos, sin una estructura definida, y sea capaz de aprender de ellos y ofrecer una respuesta en tiempo real. La conjunción de la estadística con los implacables avances en las tecnologías *Big Data* permitirán una solución a este problema.

En el Capítulo 1 presentamos los detalles teóricos del modelo FMM, así como de otros que se derivan de él, junto con información sobre la implementación del software de R desarrollado. En el Capítulo 2 extenderemos esta metodología al caso particular del ECG, un complejo caso práctico que necesita de una adaptación para ofrecer sus mejores resultados. Finalmente, en el Capítulo 3, veremos algunos ejemplos de uso de los modelos presentados anteriormente: desde patrones de luz emitidos por las estrellas, hasta ejemplos de electrocardiogramas analizados, pasando por un pequeño tutorial de utilización de la librería desarrollada.

Capítulo 1

Modelo FMM: Frecuency Modulated Möbius

Las señales oscilatorias y periódicas han estado con nosotros desde hace mucho tiempo, y los modelos que persiguen explicar su comportamiento no se han quedado atrás. El ciclo circadiano, el ciclo celular, los niveles de hormonas, y la astrofísica, son sólo un pequeño puñado de ejemplos en los que hablar de periodicidad circular en los datos es más que necesario. Por ejemplo, en las investigaciones de cronobiología que se encargan de estudiar el ciclo circadiano y el ciclo celular, entre otros, es especialmente interesante la identificación de genes cuya expresión sigue un patrón rítmico [1]. En algunos casos se han propuesto modelos paramétricos cuyo ajuste no es especialmente satisfactorio; en otros casos, algunos modelos no paramétricos han resultado de gran utilidad pero no disponen de parámetros que propongan una interpretación natural para los datos ajustados. Una discusión completa sobre datos circulares y su aplicación en cronobiología puede encontrarse en un trabajo anterior de este autor para la Universidad de Valladolid [6].

En la figura 1.1, podemos encontrar patrones rítmicos en expresiones de genes en los paneles (a) y (b). Junto a ellos, se incluye, a modo de avance, el ajuste del modelo FMM que propondremos en este trabajo en los paneles (e) y (f) respectivamente. En los paneles (c) y (d) podemos ver patrones de luces emitidas por estrellas, de nuevo junto con sus respectivos ajustes en (g) y (h).

Existen dos métodos en la literatura para el ajuste de modelos en datos que presentan una periodicidad clara, como los que se han mostrado en la figura referenciada anteriormente. Por un lado, podemos describir un modelo no paramétrico (es decir, no existen parámetros para definirlo) simplemente usando desigualdades. Se trata de modelos muy flexibles, puesto que no se basan en un modelo matemático que condicione su forma [6], [7]. Se denominan típicamente patrones Up-Down-Up, puesto que se asume que su comportamiento en un período cualquiera tiene un tramo ascendente seguido de otro descendente. Se ha demostrado que estos métodos no paramétricos basados en restricciones describen mejor que el resto de modelos paramétricos propuestos, en términos generales, estos patrones rítmicos. Sin embargo, una desventaja importante es perdemos la posibilidad de contar con una interpretación sobre los parámetros, debido

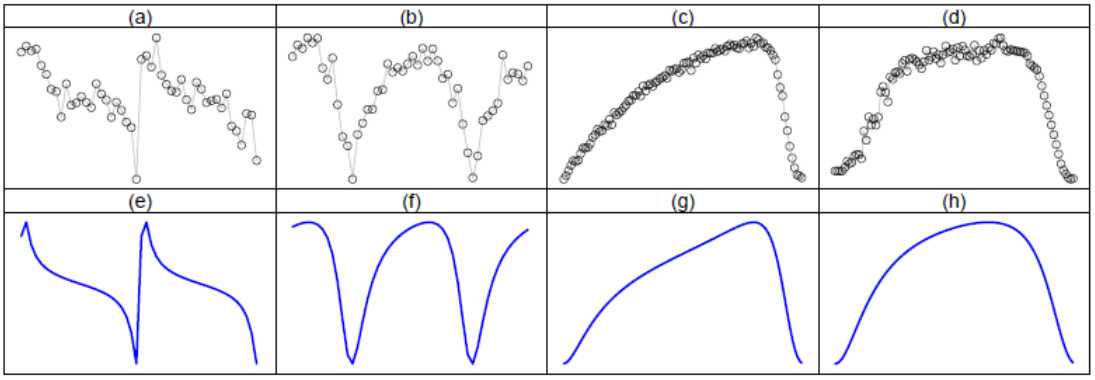


Figura 1.1: Ejemplos de expresiones de genes con dos períodos (a,b), y patrones de luz emitidos por estrellas (c,d), junto con el ajuste producido por el modelo FMM para dichos datos.

a que no existen.

El segundo grupo de métodos, al que pertenece el modelo propuesto en este trabajo, se basa en funciones matemáticas, como variaciones del coseno. Un modelo comúnmente utilizado para ajustar patrones cíclicos, aunque lejos de ofrecer resultados idóneos en entornos específicos, es el modelo Cosinor [5]. Este modelo se caracteriza por tener un parámetro de fase y otro de amplitud en cada período. Cada par de estos parámetros define una fase, y la suma de varias fases puede dar lugar a un modelo que sea capaz de representar datos circulares. De hecho, este modelo es un caso particular de una familia más conocida: la descomposición de Fourier, donde sólo se tiene en cuenta una fase.

Definición 1 (Modelo Cosinor)**.** Se define el **modelo cosinor** de la forma que sigue:

$$X(t_i) = \mu(t_i) + e(t_i) = M + A\cos(t_i + \alpha) + e(t_i),\ i = 1, \ldots, n \tag{1.1}$$

donde:

- $M \in \mathbb{R}$, $A \in \mathbb{R}^+$,
- $\alpha \in [0, 2\pi]$
- $(e(t_1), \ldots, e(t_n)) \sim \mathcal{N}(0, \sigma^2 I)$

Definición 2 (Modelo FD^m)**.** Se define el modelo FD^m de descomposición de Fourier de la forma que sigue:

$$X(t_i) = \mu(t_i) + e(t_i) = M + \sum_{j=1}^{m} [A_j \cos(n\ t_i) + B_j \sin(n\ t_i)] + e(t_i),\ i = 1, \ldots, n \tag{1.2}$$

donde:

- $M \in \mathbb{R}$, $A_j, B_j \in \mathbb{R}^+$, $j = 1 \ldots m$

- $(e(t_1), \ldots, e(t_n)) \sim \mathcal{N}(0, \sigma^2 I)$

Algunos campos, como la astrofísica, conduce ciertas investigaciones con datos circulares donde un modelo como los anteriores resulta insuficiente. Además, puede resultar interesante una interpretación automática para clasificar, por ejemplo, estrellas en función del patrón de luz que emiten. Este análisis automático pasa por desarrollar una inteligencia que sea capaz de interpretar los parámetros del modelo y, si éste no resulta adecuado y los parámetros no poseen una interpretación clara, no será posible. Esto puede ocurrir, por ejemplo, en patrones asimétricos. Se puede ajustar un modelo de descomposición de Fourier (en adelante, FD), pero requiere de muchos parámetros, lo que conduce irremediablemente al sobreajuste y problemas de interpretación.

En este capítulo exploraremos el modelo propuesto denominado *Frecuency Modulated Möbius*. Hablaremos brevemente de sus fundamentos teóricos, para centrarnos más adelante en una implementación práctica del mismo en un paquete de R, pasando por la explicación del modelo multicomponente. El contenido de este capítulo está basado en el artículo que describe el modelo publicado por el GIR *Inferencia con Restricciones* de la Universidad de Valladolid [1], además de en un artículo que está en proceso de redacción sobre su implementación en formato paquete para el lenguaje de programación *R* [2], en los que el escritor de este trabajo participa también como autor.

1.1. Modelo FMM

Motivado por las limitaciones de los modelos no paramétricos y paramétricos descritos hasta el momento, definimos a continuación las formalidades del modelo *Frecuency Modulated Möbius* (en adelante, *FMM*).

Supongamos que $X(t_i), t_1 < t_2 \ldots < t_n$ son observaciones tomadas de forma consecutiva en el tiempo. El modelo FMM se define de la forma que sigue.

Definición 3 (Modelo FMM)**.** El modelo FMM se define matemáticamente como:

$$X(t_i) = \mu(t_i) + e(t_i) = M + A\cos\left(\phi(t_i)\right) + e(t_i),\ i = 1, \ldots, n \tag{1.3}$$

donde:

- $M \in \mathbb{R}$, $A \in \mathbb{R}^+$,
- $\phi(t) = \beta + 2\text{arctan}\left(\omega \tan\left(\frac{t-\alpha}{2}\right)\right)$, $\alpha, \beta \in [0, 2\pi]$, $\omega \in [0, 1]$,
- $(e(t_1), \ldots, e(t_n)) \sim \mathcal{N}_n(0, \sigma^2 I_n)$

La justificación matemática de este modelo se puede encontrar en el material suplementario de [1]. No es el objetivo de este trabajo entrar en detalles muy concretos sobre los detalles matemáticos acerca de la adecuación de este modelo y sus orígenes.

La diferencia fundamental del modelo FMM con el modelo Cosinor es que, en lugar de utilizar simplemente la función coseno como enlace, tal y como se describe en la

definición 1, utilizamos el enlace Möbius que permite la inclusión de formas asimétricas. Este modelo consta de 5 parámetros, tan sólo 2 más que el modelo Cosinor, pero que le aportan una gran versatilidad de formas (véase la figura 1.2).

- M es el *intercept* del modelo, similar al parámetro con el mismo nombre en el modelo Cosinor.
- A es la amplitud del modelo.
- α es el parámetro de fase.
- β es un parámetro de forma que define la asimetría.
- ω es un parámetro de forma que define el apuntamiento de la señal. Si $\omega = 0$, hablamos de una señal completamente apuntada, mientras que si $\omega = 1$ trataríamos con una señal sinusoidal.

En la figura 1.2 se puede ver la influencia de los parámetros de forma para el caso en que $M = 0$, $A = 1$, y $\alpha = 0$.

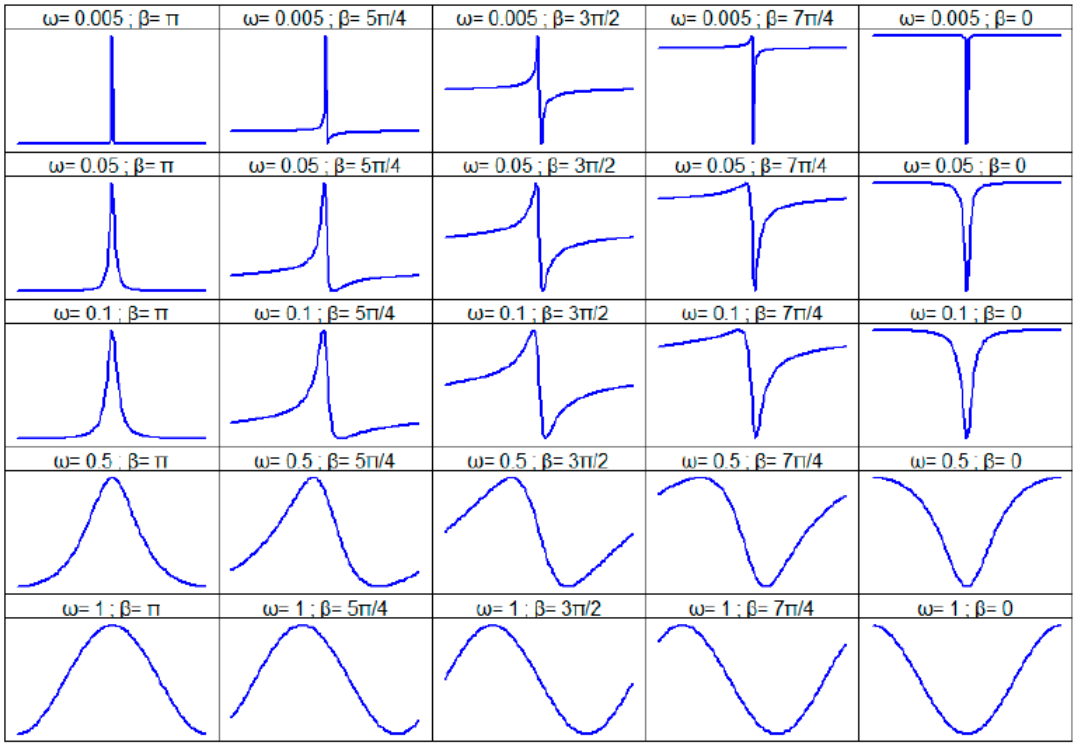


Figura 1.2: Influencia de los parámetros β y ω en un modelo FMM con $M = 0$, $A = 1$, $\alpha = 0$

Otros parámetros que son útiles y serán utilizados, sobre todo, en la interpretación automática del electrocardiograma, son los instantes de tiempo donde se alcanza el

mínimo y el máximo, t_L y t_U respectivamente. Se pueden derivar de los parámetros estimados por el modelo de la siguiente forma.

$$t_L = \alpha + 2\text{arctan}\left(\frac{1}{\omega}\tan\left(\frac{\pi-\beta}{2}\right)\right) \tag{1.4}$$

$$t_U = \alpha + 2\text{arctan}\left(\frac{1}{\omega}\tan\left(\frac{-\beta}{2}\right)\right) \tag{1.5}$$

A continuación veremos la relación entre señales circulares y oscilatorias, lo que ayudará a entender la conveniencia del FMM para el análisis de señales oscilatorias. En la sección 1.4 se abordan detalles sobre la implementación.

De ahora en adelante, supondremos que las observaciones proceden de un tiempo t en el intervalo $[0, 2\pi]$. En caso de que esta suposición no se cumpla, siempre se pueden reescalar los instantes temporales al intervalo mencionado. En primer lugar, definiremos lo que es una **señal circular**.

Definición 4 (Señal circular en el espacio euclídeo)**.** Una señal circular $\mu(t) \in \mathbb{R}$ en el espacio euclídeo es aquella en la que

- Si $t_U \leq t_L$, verifica $\mu(t) \geq \mu(t'), t_U \leq t \leq t' \leq t_L$ y $\mu(t) \leq \mu(t'), 0 \leq t \leq t' \leq t_U; t_L \leq t \leq t' \leq 2\pi$.
- Si $t_U \geq t_L$, verifica $\mu(t) \leq \mu(t'), t_L \leq t \leq t' \leq t_U$ y $\mu(t) \geq \mu(t'), 0 \leq t \leq t' \leq t_L; t_U \leq t \leq t' \leq 2\pi$.

Sin pérdida de generalidad, supondremos que $t_U \leq t_L$, también llamada de esta forma señal Up-Down-Up [6], [7].

Definición 5 (Señal circular en el círculo unidad)**.** Una señal circular $\phi(t) \in [0, 2\pi]$ lo es en el círculo unidad si y sólo si $\phi(t) \leq \phi(t')$, $0 \leq t \leq t' \leq 2\pi$.

Un ejemplo de señal circular es la señal sinusoidal procedente de un modelo cosinor, que se encuentra en la figura 1.3. Su patrón simétrico no será adecuado para representar ciertas características que sí puede hacerse con el modelo FMM (figura 1.2).

Teorema 1. Sea $\mu(t) = M + A\cos(\phi(t_i))$, con $\phi(t) = \beta + 2\text{arctan}\left(\omega\tan\left(\frac{t-\alpha}{2}\right)\right)$. Entonces:

1. $\mu(t)$ es una señal circular en el espacio euclídeo.
2. $\phi(t)$ es una señal circular en el círculo unidad.

La prueba puede consultarse en [1]. Se omiten aquí los detalles.

La importancia del resultado anterior es enorme: justifica el uso del modelo FMM en señales oscilatorias, como aquellas que resultan de los ejemplos de cronobiología o astrofísica que se utilizaron como ejemplo motivador inicial.

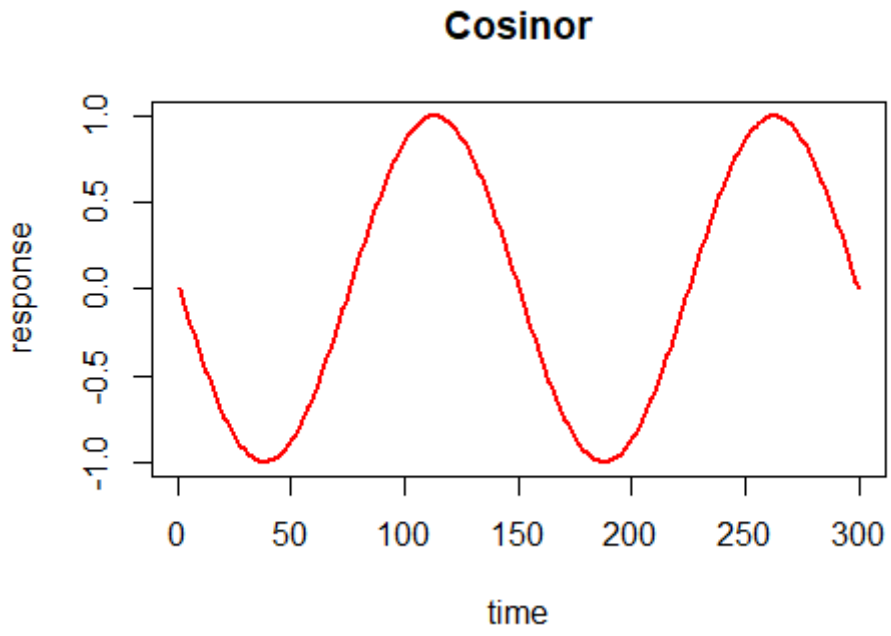


Figura 1.3: Señal generada a partir de un modelo Cosinor.

Estimación del modelo FMM

Abordaremos a continuación algunos aspectos sobre la implementación, necesarios para entender la sección 1.4 donde se abordan detalles técnicos, así como el código adjunto.

El problema de minimización al que nos enfrentamos es el siguiente:

$$\min_{\theta \in \Theta} \sum_{i=1}^{n} \left(X(t_i) - \mu(t_i, \theta) \right)^2 \tag{1.6}$$

donde θ es el hiperparámetro que contiene los parámetros expuestos anteriormente, y Θ es el espacio paramétrico.

A grandes rasgos, el método de estimación para un modelo FMM tiene dos etapas:

1. Optimización de mínimos cuadrados.
2. Optimización de Nelder-Mead.

Sea n el número de datos observados a lo largo de un período. Sea $\mathbf{X} = X(t_1), \ldots, X(t_n)$ el vector que contiene los datos observados y ordenados en el tiempo. El modelo FMM, tal y como se describe en la definición 3, asume que $\mathbf{X} \sim \mathcal{N}_n(\mu, \sigma^2 I_n)$, donde $\mu_i = M + A\cos\left(\beta + 2\arctan\left(\omega \frac{t_i - \alpha}{2}\right)\right)$ para $i = 1, \ldots, n$. Una reparametrización, que no es más que un cambio de notación, sería:

$$\mu_i = M + A\cos(t_i^* + \eta) \tag{1.7}$$

donde $t_i^* = \alpha + 2\text{arctan}\left(\omega\frac{t_i-\alpha}{2}\right)$ y $\eta = \beta - \alpha$.

Por tanto,podemos reescribir el modelo como:

$$X_i = M + \delta z_i + \gamma w_i + e_i \tag{1.8}$$

donde $\delta = A\cos(\eta)$, $\gamma = -A\sin(\eta)$, $z_i\cos(t_i^*)$, $w_i = \sin(t_i^*)$ y $e_i \sim \mathcal{N}(0, \sigma^2)$.

Consideremos unos valores fijos para α y ω. El problema se resuelve entonces a un problema resoluble por el conocido método de mínimos cuadrados, de la misma forma que el modelo Cosinor resuelve este problema. Es posible comprobar (se omiten aquí detalles, véase [1]), que los estimadores por mínimos cuadrados son:

$$\hat{M} = \overline{X} - \hat{\delta}\sum_{i=1}^{n} z_i - \hat{\gamma}\sum_{i=1}^{n} w_i \tag{1.9}$$

$$\hat{A} = \sqrt{\hat{\delta}^2 + \hat{\gamma}^2} \tag{1.10}$$

$$\hat{\beta} = \alpha + \eta \tag{1.11}$$

Durante el razonamiento anterior se han considerado valores fijos para α y ω. Para considerar las posibles variaciones de estos dos valores, se genera un *grid* con un conjunto predefinido de pares de valores. Para cada par de posibles valores de α y ω se llevan a cabo los cálculos anteriores, y se elige el mejor modelo según el criterio de máxima verosimilitud (o equivalentemente, mínimo error cuadrático medio, al hablar de un problema de mínimos cuadrados dentro del marco de un modelo Normal). Para evitar problemas de convergencia cuando ω se acerca a 0, se establece un límite para la amplitud de la onda.

Sin embargo, de esta manera estamos considerando unos valores fijos para dos parámetros, sin posibilidad de que tomen valores diferentes a los prefijados. Para evitar esto se conduce un segundo paso de optimización sobre el mejor modelo elegido, basándose en el procedimiento de optimización de Nelder-Mead. De nuevo, sin entrar en demasiados detalles matemáticos (véase [8]), se trata de una heurística utilizada para encontrar el mínimo de una función objetivo en un espacio multidimensional. Permitimos así que los parámetros se muevan en busca de minimizar el error cuadrático medio, de forma que los estimadores $\hat{\alpha}$ y $\hat{\omega}$ no provengan siempre de un grid prefijado.

En la figura 1.4 se encuentra el pseudocódigo descrito en esta sección, cuya implementación es la realizada en el paquete software desarrollado en R.

Input : $\boldsymbol{X}$, $\boldsymbol{t}$
Output: $\boldsymbol{\Theta} = (\hat{M}, \hat{A}, \hat{\alpha}, \hat{\beta}, \hat{\omega})$

#Step 1: Compute the initial estimates of $M, A, \alpha, \beta, \omega$;
$RSS \leftarrow 99999999$;
$n \leftarrow length(\boldsymbol{X})$;
$\boldsymbol{alpha} \leftarrow \text{seq}(0,2\pi,\boldsymbol{t}[2] - \boldsymbol{t}[1])$;
$\boldsymbol{omega} \leftarrow \text{seq}(0,1,\text{length.out=length}(\boldsymbol{alpha}))$;
for *a in 1:length(**alpha**)* **do**
  for *b in 1:length(**omega**)* **do**
    $\boldsymbol{t}^{\star} \leftarrow \boldsymbol{alpha}[a] + 2\arctan\{\boldsymbol{omega}[b]\tan\frac{(\boldsymbol{t}-\boldsymbol{alpha}[a])}{2}\}$;
    $CosinorReg \leftarrow lm(\boldsymbol{X} \sim \cos(\boldsymbol{t}^{\star}) + \sin(\boldsymbol{t}^{\star}))$;
    $delta \leftarrow CosinorReg\$coefficients[2]$;
    $gamma \leftarrow CosinorReg\$coefficients[3]$;
    $M\_1 \leftarrow CosinorReg\$coefficients[1]$;
    $A\_1 \leftarrow \sqrt{delta^2 + gamma^2}$;
    $alpha\_1 \leftarrow \boldsymbol{alpha}[a]$;
    $beta\_1 \leftarrow \arctan(\frac{-delta}{gamma}) + alpha\_1$;
    $omega\_1 \leftarrow \boldsymbol{omega}[b]$;
    $\textbf{MobiusReg} \leftarrow M\_1 + A\_1\cos(beta\_1 + 2\arctan(omega\_1\tan(\frac{(\boldsymbol{t}-apha\_1)}{2})))$;
    $RSS\_aux \leftarrow sum((\boldsymbol{X} - \textbf{MobiusReg})^2)/n$;
    $maxi \leftarrow M\_1 + A\_1$;
    $mini \leftarrow M\_1 - A\_1$;
    $s \leftarrow \sqrt{\frac{RSS}{n-5}}$;
    $rest1 \leftarrow maxi \leq (max(\boldsymbol{X}) + 1.96s)$;
    $rest2 \leftarrow mini \geq (min(\boldsymbol{X}) - 1.96s)$;
    if *(RSS_aux < RSS & rest1 & rest2)* **then**
      $M\_hat \leftarrow M\_1$;
      $A\_hat \leftarrow A\_1$;
      $alpha\_hat \leftarrow alpha\_1$;
      $beta\_hat \leftarrow beta\_1$;
      $omega\_hat \leftarrow omega\_1$;
      $RSS \leftarrow RSS\_aux$;
    end
  end
end
#Step 2: Conduct a Nelder-Mead optimization method to compute final estimates of $M, A, \alpha, \beta, \omega$;

$InitialParam \leftarrow c(M\_hat, A\_hat, alpha\_hat, beta\_hat, omega\_hat)$;
$ObjFunction \leftarrow function(InitialParam)\{sum((\boldsymbol{X} - \{M\_hat + A\_hat\cos(beta\_hat + 2\arctan(omega\_hat\tan(\frac{(\boldsymbol{t}-apha\_hat)}{2})))\})^2)/n\}$;
$\boldsymbol{\Theta} \leftarrow Nelder\_Mead(ObjFunction, InitialParam, rest1, rest2)$;
return $\boldsymbol{\Theta} = (\hat{M}, \hat{A}, \hat{\alpha}, \hat{\beta}, \hat{\omega})$;

Figura 1.4: Pseudocódigo para el algoritmo de estimación de los parámetros del modelo FMM

1.2. Modelo FMM multicomponente

Como acabamos de ver, el modelo FMM es un modelo de regresión no lineal cuyo abanico de formas al que se puede adaptar es muy amplio, tan sólo utilizando dos parámetros más que un modelo Cosinor. El lector puede encontrar ejemplos detallados sobre la versatilidad del modelo explicado en la sección anterior en el Capítulo 3.1.

Sin embargo, el modelo anterior sólo es capaz de encontrar un máximo y un mínimo. En muchas aplicaciones, los datos muestran patrones cíclicos donde hay más de un extremo. En este apartado se presenta un modelo FMM multicomponente diseñado para analizar datos de señales con varias oscilaciones. Este modelo ha sido por primera vez presentado en el articulo del grupo de investigación sobre el análisis de electrocardiogramas [3].

Muchas señales oscilatorias se pueden descomponer en varias componentes. Un modelo FMM con m componentes, que denotaremos como FMM_m, incluye m componentes procedentes de un modelo FMM. Presentamos a continuación su definición formal [2].

Definición 6 (Modelo FMM_m)**.** El modelo FMM_m se define matemáticamente de la forma que sigue:

$$X(t_i) = M + \mu(t_i) + e(t_i),\ i = 1, \ldots, n;\ j = 1, \ldots, m \tag{1.12}$$

donde:

- $\mu(t_i) = M + \sum_{j=1}^{m} \mu_j(t_i)$, $M \in \mathbb{R}$,
- $\mu_j(t_i) = A_j \cos(\phi_j(t_i))$, $A_j \in \mathbb{R}^+$, $j = 1 \ldots, m$,
- $\phi_j(t) = \beta_j + 2\text{arctan}\left(\omega_j \tan\left(\frac{t-\alpha_j}{2}\right)\right)$, $\alpha_j, \beta_j \in [0, 2\pi]$, $\omega_j \in [0, 1]$, $j = 1 \ldots, m$,
- $(e(t_1), \ldots, e(t_n)) \sim \mathcal{N}_n(0, \sigma^2 I_n)$

Para que el modelo sea identificable, se imponen las restricciones $\alpha_1 \leq \alpha_2 \leq \ldots \alpha_m \leq \alpha_1$ siguiendo un orden circular.

Estimación del modelo

El método de estimación que se propone para encontrar los parámetros adecuados es un procedimiento iterativo en el que se van estimando modelos monocompoenentes sucesivamente utilizando la técnica del backfitting. Fijado el número de componentes, m, y el número de iteraciones a realizar, los pasos que se siguen son los siguientes:

Para cada iteración hasta el número máximo:

1. Para cada componente desde 1 hasta m:
   a) Obtener la diferencia entre los datos reales y los ajustados por todas las demás componentes.

b) Ajustar un modelo FMM para obtener estimaciones para α_j, β_j y ω_j.

2. Ajustar un modelo de mínimos cuadrados para encontrar M y A_j, $j = 1, \ldots, m$.

3. Si se satisface el criterio de parada, detener la búsqueda.

Siguiendo estas líneas generales, el primer paso a realizar para analizar una señal que se ajusta a un modelo FMM_m sería encontrar un FMM monocomponente, siguiendo el algoritmo de la figura 1.4. De esta manera, habríamos obtenido los parámetros de la primera onda. Para obtener los de la segunda, restamos a los datos originales el ajuste de la primera y volvemos a ajustar un modelo FMM. Repetimos el proceso hasta ajustar todas las ondas y completar así la primera iteración. En la segunda iteración ya tenemos información parcial sobre el ajuste, por lo que para recalcular la primera componente, restamos de los datos observados los ajustes de todas las demás. Este proceso se repite un número de iteraciones prefijadas o hasta que se alcanzan los requerimientos de algún criterio de parada.

Como paso adicional para encontrar los parámetros M y $A_1, \ldots, A_m$, nótese que se puede escribir el modelo de forma lineal, dado que conocemos $\phi_j(t_i)$ para los distintos instantes temporales.

$$X(t_i) = M + \sum_{j=1}^{m} A_j \cos(\phi(t_i)) + e(t_i) \tag{1.13}$$

Basta por tanto un simple modelo de regresión lineal para estimar estos últimos parámetros restantes.

A modo de ejemplo con el objetivo de clarificar el funcionamiento del algoritmo de estimación, se plantea un breve ejemplo con datos simulados. En la figura 1.5 se encuentra el modelo FMM_2 sobre el que se añade un ruido procedente de una Normal para mantener las suposiciones del modelo intactas. Las dos componentes comparten los parámetros $A = 1$, $\beta = 2$ y $\omega = 0.05$. Sin embargo, difieren en la fase: $\alpha_1 = \frac{\pi}{2}$ y $\alpha_2 = \frac{3\pi}{2}$.

En la figura 1.6 (a) encontramos el resultado de ajustar un modelo FMM siguiendo el algoritmo de la figura 1.4. Tras ello, ajustamos de nuevo un modelo FMM a los residuos, obteniendo lo que se observa en el panel (b).

Finalmente, tras 4 iteraciones, en la figura 1.7 se puede comprobar el resultado final del ajuste. En el panel (a) se muestra el modelo FMM_2 completo, mientras que en (b) se muestran las dos ondas ajustadas por separado.

El algoritmo especificado arriba permite minimizar el error cuadrático medio del modelo FMM_2. El criterio de convergencia que puede utilizarse es el basado en el conocido estadístico R^2 de bondad de ajuste, o lo que es lo mismo, el porcentaje de variabilidad explicada por el modelo. Sea R_j^2 el valor de este estadístico en la iteración j. Un posible criterio de parada es cuando entre dos iteraciones consecutivas, el porcentaje de variabilidad explicada se incremente en una cantidad lo suficientemente pequeña como para considerarse insignificante, esto es, $R_j^2 - R_{j-1}^2 \leq C$. El cálculo de R_j^2 debe hacerse como sigue.

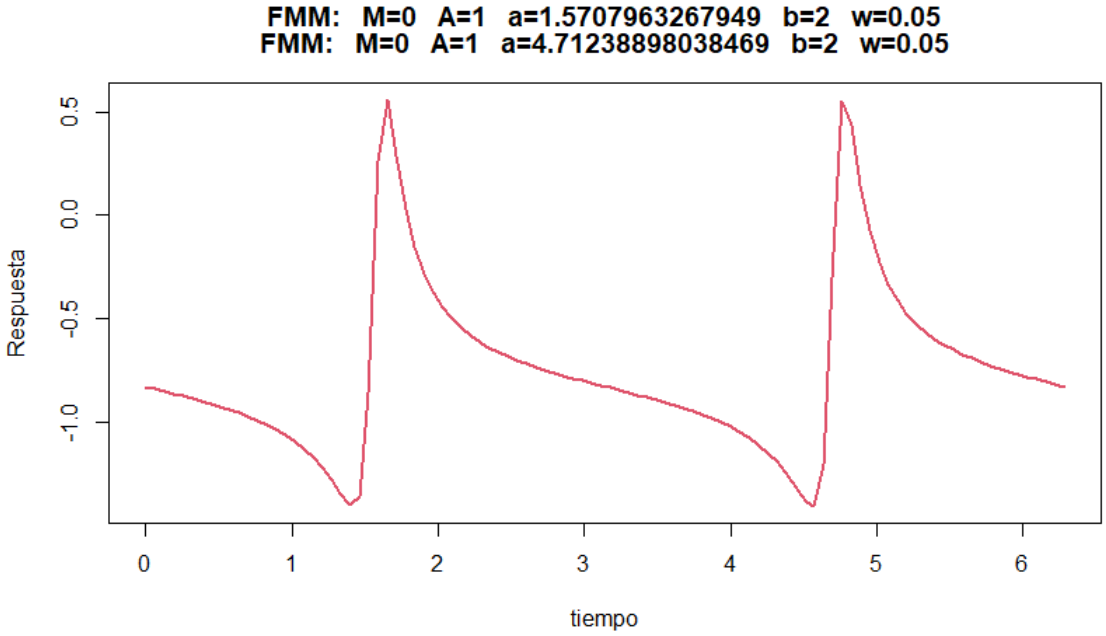
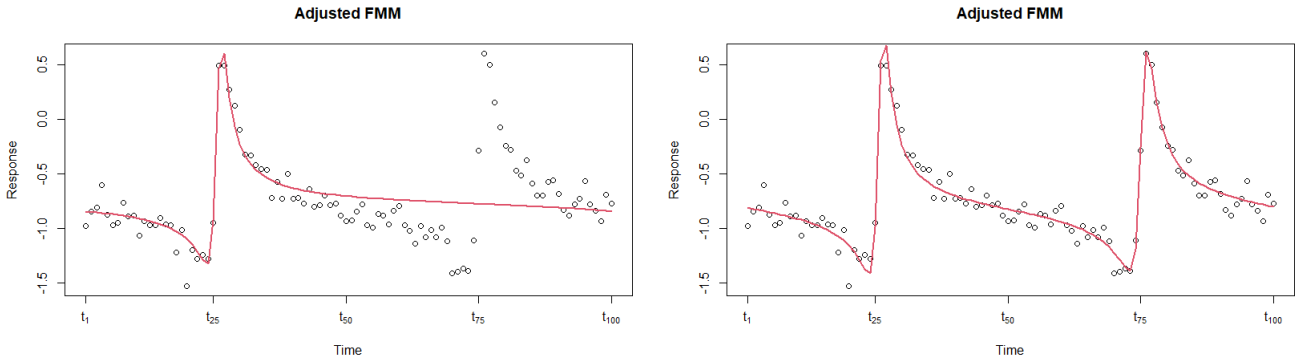


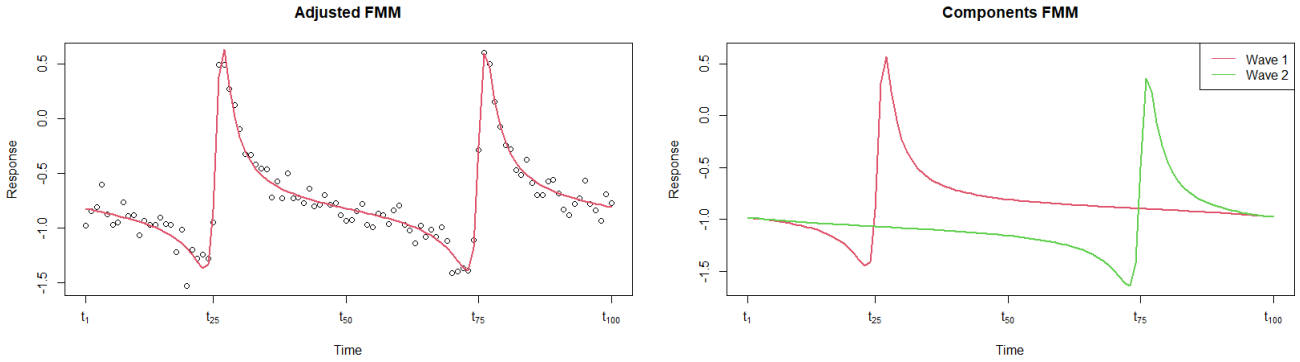
Figura 1.5: Ejemplo de datos generados bajo un modelo FMM_2



(a) Primera iteración

(b) Segunda iteración

Figura 1.6: Ajuste de la primera componente y de la segunda componente siguiendo el algoritmo de backfitting para un modelo FMM_2



(a) Ajuste final

(b) Ajuste por componentes

Figura 1.7: Ajuste final tras 4 iteraciones de datos simulados bajo un modelo FMM_2

$$R_j^2 = 1 - \frac{\sum_{i=1}^{n} \left(X(t_i) - \hat{\mu}^j(t_i)\right)^2}{\sum_{i=1}^{n} \left(X(t_i) - \overline{X}\right)^2} \tag{1.14}$$

donde $\hat{\mu}^j(t_i)$ denota el modelo ajustado FMM_m para el instante t_i en la iteración j.

Igual que ocurría en el modelo con una sola componente, resultan de gran interés los instantes temporales donde se alcanza el mínimo y el máximo. Se pueden derivar fácilmente de los parámetros de las ondas:

$$t_L^j = \alpha_j + 2\text{arctan}\left(\frac{1}{\omega_j}\tan\left(\frac{\pi - \beta_j}{2}\right)\right) \tag{1.15}$$

$$t_U^j = \alpha_j + 2\text{arctan}\left(\frac{1}{\omega_j}\tan\left(\frac{-\beta_j}{2}\right)\right) \tag{1.16}$$

Como adición al modelo FMM_m, pueden resultar de utilidad establecer ciertas restricciones sobre algunos parámetros. Trataremos este caso en la siguiente sección.

1.3. Modelo FMM multicomponente con restricciones

En el ejemplo que se ilustra en las figuras 1.5, 1.6 y 1.7 se han simulado datos procedentes de un modelo FMM_2 en el que $\beta_1 = \beta_2$ y $\omega_1 = \omega_2$. Esta situación es más común de lo que puede pensarse en un principio (véase el Capítulo 3.2 para un ejemplo más detallado). En esta sección se propone un algoritmo de estimación en la que podamos establecer restricciones adicionales a estos parámetros, que fuerzan a que la forma de las diferentes ondas sea similares en términos de asimetría o apuntamiento.

Comenzaremos hablando de restricciones sobre el parametro β. Nuestro objetivo es, bajo un modelo FMM_m, imponer las restricciones siguientes.

$$\begin{aligned}\beta_1 &= \ldots = \beta_{m_1} \\ \beta_{m_1+1} &= \ldots = \beta_{m_2} \\ &\vdots \\ \beta_{m_{d-1}+1} &= \ldots = \beta_{m_d}\end{aligned}$$

Sin entrar en demasiados detalles matemáticos (consúltese [2] para más detalles), es posible demostrar que la estimación de mínimos cuadrados verifica:

$$\begin{aligned}\hat{\beta}_j^* &= h(\hat{\beta}_1, \ldots, \hat{\beta}_{m_1}); j = 1, \ldots, m_1 \\ \hat{\beta}_j^* &= h(\hat{\beta}_{m_1+1}, \ldots, \hat{\beta}_{m_2}); j = m_1 + 1, \ldots, m_2 \\ &\vdots \\ \hat{\beta}_j^* &= h(\hat{\beta}_{m_{d-1}+1}, \ldots, \hat{\beta}_{m_d}); j = m_{d-1} + 1, \ldots, m_d\end{aligned}$$

donde $\hat{\beta}_j$ es el estimador sin restricciones para el parámetro de la onda j tal y como se explicó en el apartado anterior, y h es la función que calcula la media angular [16].

$$h(\beta_1, \ldots, \beta_n) = \text{atan2}\left(\sum_{j=1}^{n} \sin\beta_j, \sum_{j=1}^{n} \cos\beta_j\right) \tag{1.17}$$

Una vez estimados los parámetros siguiendo el algoritmo de estimación estándar del FMM_m, nuestra propuesta es simplemente forzar a cumplir las restricciones calculando la media angular de los β por los bloques que imponen las restricciones. Finalmente, se ajusta un modelo lineal nuevamente, tal y como aparece descrito en la ecuación 1.13, para determinar los estimadores de M y A.

En la misma línea, en numerosas ocasiones resulta de interés imponer restricciones sobre ω.

$$\begin{aligned}\omega_1 &= \ldots = \omega_{m_1} \\ \omega_{m_1+1} &= \ldots = \omega_{m_2} \\ &\vdots \\ \omega_{m_{d-1}+1} &= \ldots = \omega_{m_d}\end{aligned}$$

En este caso el problema es más complicado, puesto que ω es un parámetro que se establece "a la fuerza" por medio de un grid en cada ajuste FMM individual. Eso significa que, a priori, las llamadas sucesivas a la función de ajuste unitaria FMM descrita en la figura 1.4 devolvería un ω diferente en cada caso. Además, no existe un

resultado teórico que demuestre que la estimación de mínimos cuadrados se resuelve a una mera operación matemática para encontrar los estimadores óptimos, como sí ocurre con los β.

En este caso se plantearán diferentes opciones. La más completa, pero también costosa, es extraer el grid de ω fuera del bucle de la estimación. En este caso, lo que haríamos sería ajustar diferentes modelos FMM_m con unos valores para ω prefijados que satisfagan las restricciones impuestas anteriormente. Por ejemplo, si la restricción es que $\omega_1 = \ldots = \omega_m$, es decir, que sólo haya un bloque de ω ($d = 1$), la solución es sencilla. Probaríamos diferentes modelos FMM_m, imponiendo en cada uno de ellos unos cuantos valores para ω, y luego elegiríamos el mejor. Finalmente, en un paso de optimización similar a la optimización de Nelder-Mead descrita anteriormente [8], podríamos permitir variaciones en la estimación de este parámtro.

La realidad es practicable si $d = 1$, pero cuando hay varios bloques de ω, se hace necesario explorar todas las combinaciones posibles, y la complejidad temporal necesaria no dejaría de crecer. De hecho, es del orden $O(n^{2d})$. Una de las aplicaciones de este trabajo es el análisis de grandes cantidades de flujos de datos procedentes de un electrocardiógrafo, tal y como se explicará en el Capítulo siguiente, por lo que no resulta factible.

Se proponen a continuación dos opciones para solventar este problema que, si bien no encuentran la solución óptima, constituyen una heurística muy eficaz para atajar el problema.

La primera opción es similar a la propuesta para β, con la salvedad de que la información debe ser refrescada en cada iteración y no sólo al final. Consideramos los estimadores sin restricciones para ω tras la iteración s, de forma que $\hat{\omega}_j^{(s)}$ es la estimación para el parámetro en la onda j. Una vez calculados todos, y antes de pasar a la siguiente iteración, se modifican de la forma que sigue:

$$\begin{aligned} \hat{\omega}_j^{(s)*} &= mean(\hat{\omega}_1^{(s)}, \ldots, \hat{\omega}_{m_1}^{(s)}); j = 1, \ldots, m_1 \\ &\vdots \\ \hat{\omega}_j^{(s)*} &= mean(\hat{\omega}_{m_{d-1}+1}^{(s)}, \ldots, \hat{\omega}_{m_d}^{(s)}); j = m_{d-1}+1, \ldots, m_d \end{aligned}$$

La otra posibilidad pasa por considerar los estimadores iniciales de los bloques, $\hat{\omega}_1^{(s)*}, \ldots, \hat{\omega}_{m_{d-1}+1}^{(s)}$ y definir los demás tal que

$$\begin{aligned} \hat{\omega}_j^{(s)*} &= \hat{\omega}_1^{(s)}; j = 1, \ldots, m_1 \\ &\vdots \\ \hat{\omega}_j^{(s)*} &= \hat{\omega}_{m_{d-1}+1}^{(s)}; j = m_{d-1}+1, \ldots, m_d \end{aligned}$$

Cualquiera de las dos opciones planteadas no lleva a la solución óptima, pero resulta en unas estimaciones más que aceptables.

1.4. Implementación de un paquete en R

Uno de los principales objetivos de este trabajo es la creación de un paquete, programado en R, que permita al público en general ajustar los modelos anteriormente propuestos. Junto con la creación de este repositorio de funciones de ajuste, se está preparando en paralelo un artículo, dirigido a la revista *Journal of Statistical Software*, donde se describen los detalles del modelo y su implementación. Está previsto que el repositorio pueda descargarse del CRAN oficial de R a finales de 2020.

Todas las funciones necesarias para ajustar los modelos FMM mencionados anteriormente están incluidas en este paquete, así como otras funciones de utilidad para facilitar su interpretación y comprensión. Adicionalmente, se presenta junto a este trabajo los ficheros fuente que permiten la instalación del paquete de forma local, así como todo el código original que contiene. En la ruta relativa *FMM/R/* (partiendo desde el directorio donde se encuentra este archivo) podrá encontrar todo el código que forma parte del repositorio.

En esta sección trataremos los detalles técnicos de su implementación. Para ejemplos de uso, el lector puede dirigirse a la sección 3.3. En esta sección exploramos, en primer lugar, la utilización de la función principal de ajuste. Esta función devuelve un objeto de tipo FMM, en el que entraremos en detalles, creado para almacenar información relevante sobre el modelo ajustado. Finalmente, exploraremos otras funciones de utilidad que contiene el paquete.

1.4.1. Función de ajuste

Existen tres funciones principales que realizan el ajuste, en función de las necesidades:

- *fitFMM_unit*: realiza el ajuste de un modelo FMM monocomponente.
- *fitFMM_back*: realiza el ajuste de un modelo FMM_m.
- *fitFMM_restr*: realiza el ajuste de un modelo FMM_m con las restricciones impuestas sobre los parámetros.

Sin embargo, siguiendo los principios de encapsulación y transparencia de cara al usuario final, este paquete ofrece una sola interfaz de entrada, a través de la función *fitFMM*. Internamente, la llamada se dirige hacia cada uno de los casos que más convenga para realizar el ajuste.

Esta función de ajuste principal, *fitFMM*, contiene una amplia lista de argumentos con los que se puede parametrizar el modelo. Hablaremos en primer lugar de estos argumentos. La lista completa es la siguiente.

- *vData*: contiene los datos para el ajuste, $X(t_1), \ldots, X(t_n)$.

- *numPeriods*: contiene el número de períodos de los datos. Es posible utilizar esta función para ajustar datos de más de un período. En este caso, se calcula la media por períodos, y se ajusta el modelo FMM utilizando esto como resumen. Por defecto, toma el valor 1.

- *timePoints*: contiene los instantes de tiempo en los que se observan los datos, $t_1, \ldots, t_n$. Por defecto, están equiespaciados entre 0 y 2π, pero el programa soporta el caso general donde el paso no es el mismo en todos los puntos temporales.

- *nback*: es el valor m que contiene el número de componentes del modelo FMM_m. Por defecto se toma el valor 1.

- *betaRestrictions*: permite incorporar la información sobre las restricciones de β. Se trata de un vector numérico donde dos números iguales se refieren a una restricción de igualdad sobre los β correspondientes. Por ejemplo, si consideramos $m = 4$ y el vector $(1, 1, 2, 2)$, estamos imponiendo las restricciones $\beta_1 = \beta_2$ y $\beta_3 = \beta_4$. Por defecto no se imponen restricciones.

- *omegaRestrictions*: permite incorporar al modelo información sobre las restricciones de ω, en el mismo formato que el anterior. Por defecto no se imponen restricciones.

- *maxiter*: número máximo de iteraciones que se realiza en la búsqueda del ajuste FMM_m.

- *stopFunction*: función que se utiliza como criterio de parada. Existen dos funciones implementadas en la librería: *alwaysFalse* y *R2*. La primera, utilizada por defecto, nunca provocará la parada en la búsqueda y dejará que se alcance el máximo número de iteraciones. La segunda detiene la búsqueda si el porcentaje de variabilidad explicada entre dos iteraciones consecutivas es menor del 0.1 %. Es posible parametrizar esta última función con el parámetro *difMax*, aunque se ha comprobado experimentalmente que este es un buen valor. En el fichero *stopFunctions.R* dentro de la ruta mencionada previamente pueden encontrarse los detalles.

- *lengthAlphaGrid*: contiene la precisión a utilizar en el grid del parámetro α. Como se detalla en la figura 1.4, se utiliza un grid tanto para α como para ω en el primer paso del ajuste. A mayor precisión, es decir, número de valores diferentes del parámetro que se probarán, mayor será la bondad del ajuste obtenido, pero el tiempo de ejecución crecerá de forma cuadrática con esta precisión. Por defecto toma el valor 48.

- *lengthOmegaGrid*: contiene la precisión a utilizar en el grid del parámetro ω. Por defecto, vale 24.

- *numReps*: contiene el número de búsquedas que se realizan haciendo "zoom" sobre el grid. Este paso añade una capa adicional al algoritmo presentado en la figura 1.4. Sin entrar en demasiados detalles, es posible, una vez seleccionado el mejor

ajuste partiendo de un grid predefinido, realizar otro ajuste posterior con otro grid diferente, pero utilizando una ventana cercana a los valores estimados por el ajuste anterior. El número de veces que se repite este ajuste está gobernado por este parámetro, que por defecto toma el valor 3.

- *showProgress*: un *flag* que permite ir mostrando por pantalla el progreso realizado por la función de ajuste. Por defecto está activado.
- *showTime*: un *flag* que permite mostrar en pantalla el tiempo invertido por la función de ajuste, una vez que ésta ha terminado. Por defecto está activado.
- *parallel*: un *flag* que permite paralelizar la búsqueda en distintos procesadores. Puede mejorar los tiempos de ejecución en caso de que el modelo que se requiere ajustar sea demasiado complejo.

La función redirige la llamada a una de las que realizan el ajuste siguiendo el siguiente esquema:

1. Si $m = 1$, entonces se utiliza la función *fitFMM_ unit*.
2. Si $m \neq 1$, pero no se establecen restricciones sobre parámetros, se utiliza *fitFMM_ back*.
3. En otro caso, se utiliza *fitFMM_ restr*.

En la ruta donde se encuentra el código fuente se puede consultar el contenido exacto de estas funciones. Existe un archivo para cada una de ellas.

Una vez que la función de ajuste ha terminado, devuelve como resultado un objeto de tipo FMM, que puede ser interpretado por el usuario o a través de las demás funciones de utilidad proporcionadas por el paquete.

1.4.2. Objeto de tipo FMM

El objeto FMM es un objeto de tipo $S4$ dentro de R. Los campos que son necesarios en dicho objeto para almacenar toda la información del ajuste son los siguientes:

- M: se trata de un valor numérico que contiene el valor estimado para M (véase la definición 6 del modelo FMM_m).
- A: vector numérico que contiene la estimación de $A_1, \ldots, A_m$.
- *alpha*: vector numérico que contiene la estimación de $\alpha_1, \ldots, \alpha_m$.
- *beta*: vector numérico que contiene la estimación de $\beta_1, \ldots, \beta_m$.
- *omega*: vector numérico que contiene la estimación de $\omega_1, \ldots, \omega_m$.
- *nPeriods*: contiene el número de períodos de los datos almacenados.

- *timePoints*: contiene los instantes de tiempo $t_1, \ldots, t_n$.
- *summarizedData*: contiene los datos $X(t_1), \ldots, X(t_n)$ tras aplicar el proceso de sumarización, en caso de existir más de un período.
- *adjusted*: contiene los datos ajustados por el modelo, $\hat{\mu(t_i)}$.
- *SSE*: error cuadrático medio del ajuste.
- *data*: datos en crudo, tal cual han sido recibidos por la función de ajuste.
- *R2*: porcentaje de variabilidad explicada por el modelo.

El usuario puede imprimir el contenido de este objeto por pantalla, así como utilizar las diferentes funciones de utilidad que se describen a continuación.

1.4.3. Otras funciones de utilidad

Adicionalmente a la función de ajuste, sin lugar a dudas la estrella del paquete desarrollado, se adjuntan una serie de funciones destinadas a facilitar la interpretabilidad o a la extracción de conclusiones y de información de interés. En este breve apartado describiremos tres adiciones al software ya descrito: la función de simulación, la función de representación, y las funciones heredadas de modelos de regresión.

La función dedicada a la representación de resultados pretende facilitar la tarea de representar la información junto con el modelo que ha sido ajustado. Son dos los gráficos que se pueden generar utilizando esta función: el gráfico de ajuste, y el gráfico de componentes. Éste último sólo tiene sentido al hablar de un modelo FMM_m. Los argumentos de esta función son los siguientes.

- *objFMM*: el objeto FMM generado por la función de ajuste, que contiene la información necesaria para la representación.
- *components*: valor *booleano* que define el tipo de gráfico. Por defecto se encuentra a *false*, lo que indica que el gráfico que se construirá es un gráfico de ajuste completo del modelo, superpuesto sobre los datos. Este gráfico se encuentra en la parte izquierda de la figura 1.8. Si se establece a *true*, el gráfico generado será el de componentes. En él se superponen las distintas componentes (normalizadas para que todas comiencen y terminen en el mismo punto), y se puede apreciar de forma clara la contribución de cada componente al ajuste total. Esto se representa a la derecha de la figura 1.8.
- *plotAlongPeriods*: para datos sobre más de un período, puede establecerse el valor *true* para representar todos ellos. Por defecto, el valor es *false*.
- *use_ggplot2*: permite el uso de una librería más avanzada para la generación de gráficos, *ggplot2*, de forma opcional. Por defecto, el valor es *false*.
- *legendInComponentsPlot*: en caso de estar activado, incluye una leyenda en el gráfico de componentes. Por defecto, el valor es *true*.

- *textExtra*: un *string* que permite la inclusión de texto extra en el título de la representación.

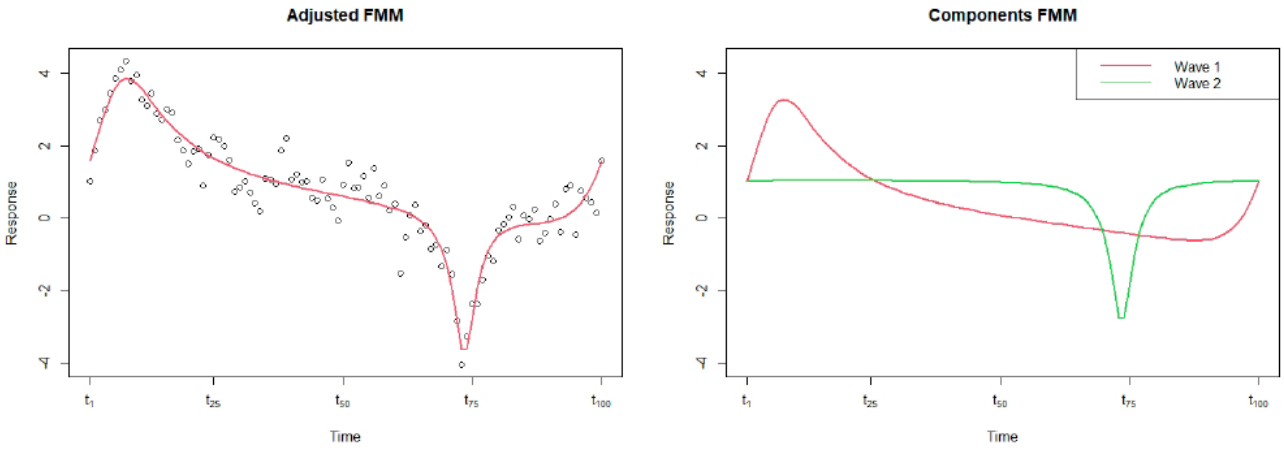


Figura 1.8: Ejemplo de salida de las dos posibles representaciones gráficas de la función plotFMM

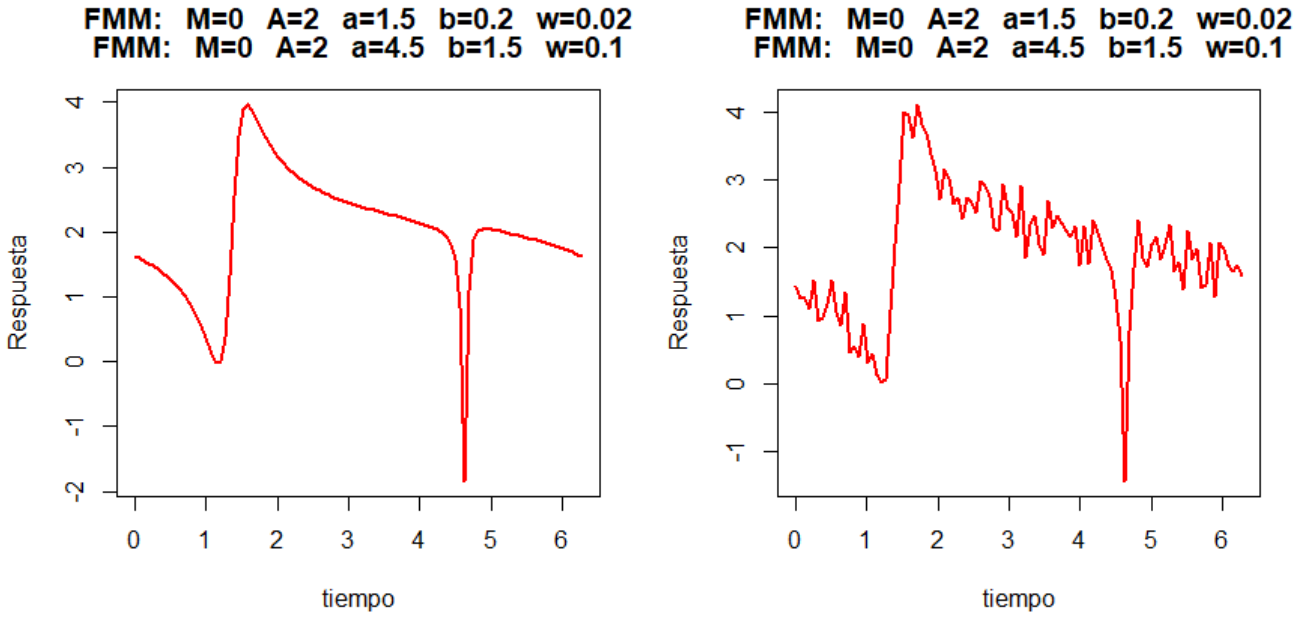
Otra función de interés es la de simulación. Mediante su utilización, se generan datos aleatorios que siguen las suposiciones de un modelo FMM_m: señal oscilatoria más un ruido gaussiano. La función se llama *generate_FMM* y tiene los siguientes posibles argumentos:

- M: Vector numérico con los valores de M para las distintas ondas.
- A: Vector numérico con los valores de A.
- *alpha*: Vector numérico con los valores de α.
- *beta*: Vector numérico con los valores de β.
- *omega*: Vector numérico con los valores de ω.
- *from*: Instante de inicio de la generación de oobservaciones. Por defecto, 0.
- *to*: Instante de fin de la generación de oobservaciones. Por defecto, 2π.
- *length.out*: longitud del vector de observaciones generadas. Por defecto, 100.
- *timePoints*: Vector numérico que contiene los instantes de tiempo en los que se observan los datos simulados. Por defecto, son instantes equiespaciados entre *from* y *to*.
- *plot*: Variable *flag* que indica si se muestra una representación de los datos simulados o no. Por defecto está activada.
- *outvalues*: Variable *flag* que indica si se devuelven los resultados numéricos de la simulación. Por defecto está desactivada.

- *sigmaNoise*: Valor del ruido gaussiano añadido a la señal (desviación estándar). Por defecto, no se añade ruido.

En caso de que la longitud de los valores numéricos para los parámetros del modelo FMM_m no coincidan, se extenderán hasta la longitud máxima por replicación.

En la figura 1.9 se encuentran un par de salidas gráficas de datos simulados con esta función. Todos los parámetros se han dejado por defecto en el panel (a), mientras que en el (b) se ha añadido un pequeño ruido gaussiano.



(a) Sin ruido

(b) Con ruido ($sigmaNoise = 0.3$)

Figura 1.9: Datos generados con la función de generación generate_FMM

Finalmente, a semejanza de otros métodos de R para el ajuste de modelos de regresión, se incluyen las siguientes funciones:

- *summary*: muestra por pantalla un resumen del modelo ajustado, así como de su bondad de ajuste.
- *coef*: permite extraer los coeficientes (parámetros) estimados del modelo.
- *fitted*: devuelve los valores ajustados por el modelo para los instantes temporales especificados.
- *predict*: permite elaborar predicciones para nuevos instantes temporales.

Se mantienen los mismos nombres para ayudar al principio de transparencia y de encapsulamiento de código, de forma que para el usuario final, los nombres y la utilización de estas funciones son los mismos como para cualquier otro modelo de regresión.

Para consultar más información sobre este aspecto de las funciones de regresión de R, consúltese la ayuda de la librería *stats* de dicho lenguaje de programación ([9]).

Capítulo 2

Modelo FMM ECG

La importancia de una señal de electrocardiograma (ECG de ahora en adelante) en tareas de diagnosis y anticipación de enfermedades cardiovasculares es enorme. La señal grabada por un electrocardiógrafo refleja directamente la actividad eléctrica del corazón: contracción y relajación de las cavidades auriculares y ventriculares, sus despolarizaciones y repolarizaciones. Finalmente, esta actividad se plasma en una figuras en 2D, cada una de ellas constando típicamente de 5 ondas, etiquetadas como P, Q, R, S y T [10]. La teoría detrás de la captura, análisis e interpretación de un ECG es gigantesca, por lo que incluiremos aquí únicamente el conocimiento del dominio mínimo e imprescindible para la comprensión del modelo propuesto.

Típicamente, la actividad de un ECG se recoge mediante 12 electrodos colocados estratégicamente en distintas posiciones del cuerpo. La actividad eléctrica en el corazón es, como cabe esperar, tridimensional, puesto que no es un órgano plano. El electrocardiógrafo produce datos en 12 derivaciones diferentes, que no es más que una forma de capturar la información en 3D sobre proyecciones en distintos planos. Desde el punto de vista de la información almacenada, existe mucha redundancia en la información al almacenarla de esta manera. Desde el punto de vista más práctico, están elegidas de forma que los profesionales sanitarios puedan identificar patologías de la forma más óptima, puesto que algunas se ven mejor en algunas derivaciones, y otras en diferentes proyecciones. La derivación más famosa es la número 2, y es en la que nos centraremos en el presente trabajo.

En este capítulo se expone una nueva aproximación basada en el modelo FMM_m para el análisis de un ECG, además de acompañarlo de los detalles y el código de su implementación. Por sus orígenes, se ha bautizado como FMM_{ECG}, inicialmente propuesto por parte del grupo de investigación en [3]. En la figura 2.1 se encuentran las 5 ondas típicas de un latido común, procedentes del ajuste de un modelo FMM_{ECG}. En la figura 2.2 se encuentran los datos reales extraídos de un electrocardiógrafo (fuente: Physionet), junto con su ajuste del modelo FMM_{ECG}.

Hasta ahora, la mera observación de un ECG por parte de los cardiólogos más experimentados es la forma de diagnosis de enfermedades cardiovasculares, desde la arritmia cardíaca hasta un inminente infarto de miocardio. No hace falta remarcar lo obvio: la cantidad de recursos humanos que requiere este procedimiento es enorme, pues

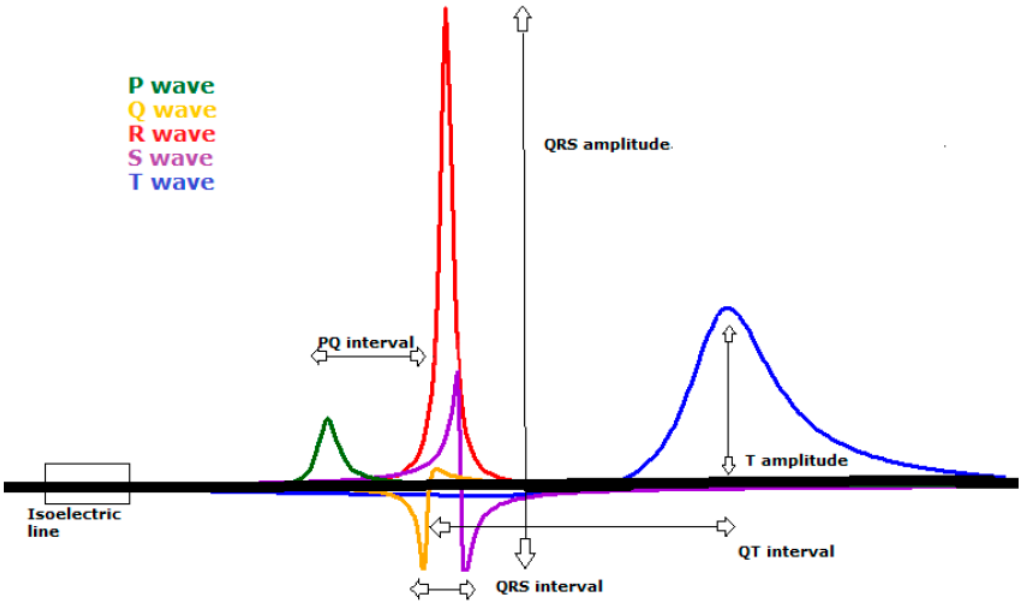


Figura 2.1: ondas P, Q, R, S y T extraídas de un ajuste del modelo FMM_{ECG}.

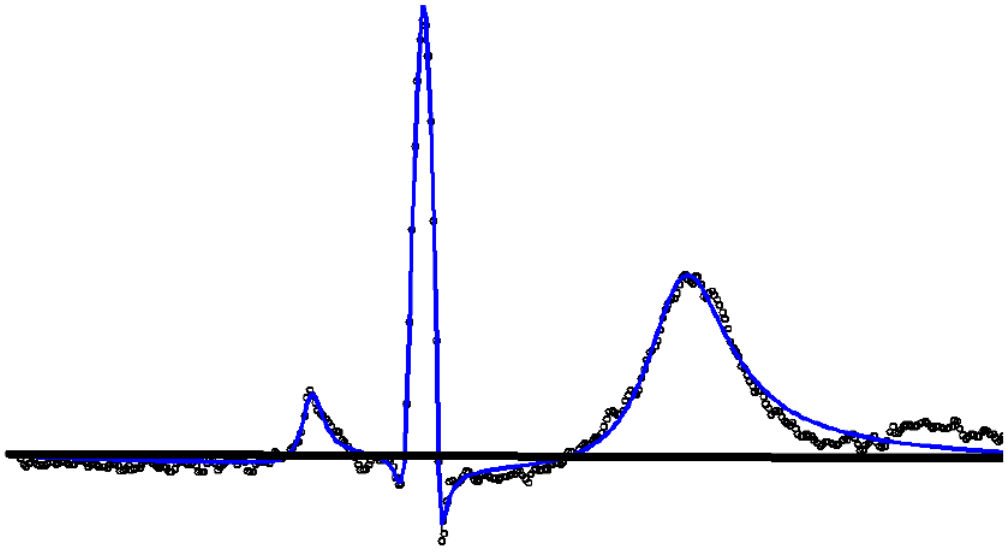


Figura 2.2: Datos reales de un ECG procedentes de Physionet (paciente sel106) (www.physionet.org) junto con su correspondiente ajuste del modelo FMM_{ECG}

parte de un minucioso análisis por un experto cardiólogo. Además, no existe hasta ahora un método automático fiable que permita realizar el diagnóstico de forma precisa.

La cantidad de métodos que se han propuesto, con una tasa de acierto más que cuestionable, es inmensa. Podrían distinguirse dos grandes tipos de métodos: técnicas de aprendizaje automático, y modelos matemáticos interpretables. Éstos últimos han apostado por modelos que resultan de combinación de funciones gaussianas para ajustar un latido. Sin embargo, no se han demostrado efectivas para expresar la morfología de un latido, especialmente cuando existe ruido o alguna patología visible en la señal. Puede encontrarse más información de esta familia de métodos en [11].

Por otro lado, las aproximaciones más computacionales y de *machine learning* son las que se encuentran más en auge. En general, son bastante dependientes del conjunto de entrenamiento, la selección de grupos de diagnóstico, y el preprocesamiento realizado. En muchos casos funcionan como cajas negras en un campo donde una amenaza inminente debe ser justificada de forma comprensible. Algunas aproximaciones de este estilo pueden verse en [12], [13].

La aproximación propuesta en este trabajo, FMM_{ECG}, es una basada en modelos matemáticos. Hablamos de un modelo FMM_5 en esencia (aunque el número de componentes puede sufrir variaciones, como veremos más adelante). Cada una de las cinco ondas se representa con 4 parámetros, que son suficientes para describir su amplitud, localización, asimetría y apuntamiento. Para estimar estos parámetros, se propone un algoritmo de Maximización-Identificación (MI). Recordamos que la metodología presentada a continuación se ha propuesto en la literatura muy recientemente [3].

El algoritmo MI propuesto comienza con un paso de maximización de la verosimilitud. En el siguiente paso, se trata de identificar las diferentes ondas en función de sus parámetros, y asignarles una etiqueta P, Q, R, S y T. Estos dos pasos se alternan iterativamente. Cabe destacar que el paso I depende de la derivación utilizada, aunque puede adaptarse manualmente a las demás.

Son cinco las principales ventajas del modelo FMM_{ECG}, que enumeramos a continuación.

- El modelo FMM_{ECG} contiene información física acerca de la conducción eléctrica de un latido. Cada una de las cinco ondas representa una parte de la despolarización o repolarización auricular o ventricular.
- Cada onda queda definida con cuatro parámetros que aportan información sobre la amplitud, localización, escala y forma. Permiten caracterizar, reproducir e identificar la enorme variedad de morfologías que se encuentran en un ECG. Además, otras características de interés se pueden derivar de estos parámetros, como el instante temporal donde se alcanza el mínimo o el máximo.
- El algoritmo MI alcanza estimaciones precisas y robustas evitando problemas de sobreajuste.
- La aproximación no depende de un conjunto de entrenamiento y es válido para cualquier señal ECG, independientemente de su preprocesamiento.

- Tiene buenas propiedades demostrables: es la de máxima verosimilitud suponiendo error Normal, los parámetros son identificables, y los estimadores son consistentes.

El gran reto es utilizar la aproximación descrita en este Capítulo como una tarea de análisis de grandes flujos de datos. El procedimiento es computacionalmente intensivo, e idealmente sería capaz de preprocesar y analizar los latidos siguiendo esta aproximación, para dar un diagnóstico automático fiable en tiempo real. Esto, sin embargo, es más complicado de lo que puede parecer, tal y como veremos en la sección dedicada a los detalles de implementación.

2.1. Modelo FMM ECG

Supongamos que $X(t_i)$, $t_1 < \ldots < t_n$ las observaciones de un latido. Sin pérdida de generalidad, supondremos que $t_i \in [o, 2\pi]$. En cualquier otro caso, los instantes temporales pueden ser reescalados a este intervalo.

Para $J \in \{P, Q, R, S, T\}$ sea $v_J = (A_J, \alpha_J, \beta_J, \omega_J)$ el hiperparámetro que describe la forma de cada una de las ondas, de forma que

$$W_J(t, v_j) = A_j \cos\left(\beta_j + 2atan\left(\omega_J \tan\left(\frac{t - \alpha_j}{2}\right)\right)\right) \tag{2.1}$$

Presentamos a continuación la definición formal del modelo FMM_{ECG}.

Definición 7 (Modelo FMM_{ECG})**.** El modelo FMM_{ECG} se define matemáticamente de la forma que sigue:

$$X(t_i) = \mu(t_i, \theta) + e(t_i) \tag{2.2}$$

para $i = 1, \ldots, n$, donde

$$\mu(t, \theta) = M + \sum_{J \in \{P,Q,R,S,T\}} W_j(t, v_j) \tag{2.3}$$

y además

- $\theta = (M, v_P, v_Q, v_R, v_S, v_T)$ verificando:
  1. $M \in \mathbb{R}$.
  2. $v_J \in \mathbb{R}^+ \times [0, 2\pi] \times [0, 2\pi] \times [0, 1]$, $J \in \{P, Q, R, S, T\}$.
  3. $\alpha_P < \alpha_Q < \alpha_R < \alpha_S < \alpha_T < \alpha_P$ siguiendo un orden circular.
- $(e(t_1), \ldots, e(t_n)) \sim \mathcal{N}_n(0, \sigma^2 I_n)$

La incorporación del orden circular como restricciones para α representa el movimiento ordenado de los estímulos eléctricos desde el extremo superior izquierdo, comenzando por la despolarización auricular, hasta su propagación a las cavidades ventriculares terminando con su repolarización. Esto dota al modelo de una interpretabilidad,

puesto que fuerza el hecho de que las ondas se sucedan en el orden correcto. Además, garantiza la identificabilidad de los parámetros.

Otra característica de interés son los llamados *fiducial points* definidos para $J \in \{P, Q, R, S, T\}$ de la forma que sigue.

$$t_L^J = \alpha_J + 2\text{arctan}\left(\frac{1}{\omega_J}\tan\left(\frac{\pi - \beta_J}{2}\right)\right) \tag{2.4}$$

$$t_U^J = \alpha_J + 2\text{arctan}\left(\frac{1}{\omega_J}\tan\left(\frac{-\beta_J}{2}\right)\right) \tag{2.5}$$

El coeficiente de determinación que mide la proporción de variabilidad explicada es

$$R^2 = 1 - \frac{\sum_{i=1}^n \left(X(t_i) - \hat{\mu}(t_i)\right)^2}{\sum_{i=1}^n \left(X(t_i) - \overline{X}\right)^2} \tag{2.6}$$

2.2. Implementación en R

El algoritmo que se describe en esta sección está especialmente diseñado (aunque puede adaptarse muy fácilmente a otros conjuntos de datos) para el ajuste de la base de datos QT de Physionet (www.physionet.org). En esta base de datos se proporcionan las anotaciones QRS, es decir, está indicado dónde se encuentra el complejo QRS, identificando principalmente la onda R, responsable del mayor movimiento eléctrico del corazón. Éste es un problema conocido y resuelto, por lo que supondremos que el segmento ECG que llega al algoritmo tiene el necesario preprocesamiento ya realizado [14].

Concretamente, el algoritmo implementado recibe como entradas:

- Datos numéricos del ECG que se suponen equiespaciados temporalmente.
- Instante de inicio y fin del latido.
- Anotación del pico de la onda R.

El algoritmo propuesto responde al siguiente problema de optimización

$$\min_{\theta \in \Theta} \sum_{i=1}^n \left(X(t_i) - \mu(t_i, \theta)\right)^2 \tag{2.7}$$

donde Θ es el espacio paramétrico.

Computacionalmente hablando, no resulta una tarea sencilla. Debemos alternar entre el paso M de maximización para encontrar las mejores ondas, y el paso I de asignarles una letra. A grandes rasgos, la aproximación es similar al *backfitting* propuesto en el modelo FMM_m, pero con el paso de identificación añadido al final de cada iteración.

Al final del paso M resultan, al menos, 5 componentes oscilatorias siguiendo el algoritmo propuesto para el FMM_m. El paso I asigna letras a, como máximo, 5 de las componentes encontradas en el paso M. Lo habitual es que sean 4, aunque bajo la presencia de ruido considerable o cuando la morfología de la onda es patológica, a veces, las verdaderas ondas quedan escondidas en la sexta y séptima componente (rara vez en las siguientes).

Para cada componente se determina el porcentaje de variabilidad que contribuye a explicar, PV. El porcentaje de variabilidad asociado a una onda k se define como

$$PV_k = R^2_{1,\ldots,k} - R^2_{1,\ldots,k-1} \tag{2.8}$$

donde $R^2_{1,\ldots,k}$ se refiere al coeficiente de determinación en un modelo FMM_k.

En algunas ocasiones donde la morfología del ECG no sea la habitual, será necesario emplear algunos thresholds establecidos tras un estudio minucioso del dominio y de los ECG publicados en Physionet. Al inicio de cada iteración, mantendremos información parcial de aquellas ondas cuya forma coincida con alguna de las 5 ondas posibles para el ECG (P, Q, R, S y T). El algoritmo termina cuando el valor de R^2 se incrementa en menos de un 0.01 % con respecto a la iteración anterior, o cuando se llega a un máximo de 10 iteraciones.

A continuación describimos con un poco más de detalle los dos pasos.

1. En el **paso M**, se utiliza el algoritmo de ajuste del modelo FMM_5 para encontrar el ajuste con 5 componentes.

2. En el **paso I** se lleva a cabo la asignación de las ondas. La onda R se asigna en primer lugar, utilizando como referencia la anotación de QT, y verificando algunas restricciones de integridad: $\omega_R < 0.12$ (debe ser una onda apuntada) y que sea el máximo valor de todo el ECG (que $\mu(t^U_R)$ sea el máximo global). A continuación, se lleva a cabo una preasignación de las componentes P, Q, S y T utilizando las restricciones sobre α: $\alpha_P < \alpha_Q < \alpha_R < \alpha_S < \alpha_T < \alpha_P$. Los pasos sucesivos se necesitan cuando la preasignación no es suficiente por ser un ECG que se aleja de lo corriente. Puede deberse a la ausencia de ondas, a una patología, o a la presencia de mucho ruido. En este caso, se llevan a cabo diferentes asignaciones utilizando thresholds desarrollados gracias a la introducción del conocimiento de determinados expertos en la materia. Por ejemplo, en ocasiones se puede confundir una onda P con una Q. Suponiendo que $\alpha_P < \alpha_Q < \alpha_R$, Q es normalmente más apuntada, por lo que debe cumplirse que $\omega_Q < \omega_P$.

La salida del paso I se considera satisfactoria cuando las cinco ondas han sido encontradas y asignadas y exhiben una morfología que cumple ciertas restricciones de integridad.

En la figura 2.3 se encuentra un esquema gráfico de los pasos que sigue el algoritmo de estimación.

Adicionalmente, en la ruta relativa *FMMECG/* se presenta el código que implementa este procedimiento completo. A continuación mostramos únicamente una descripción

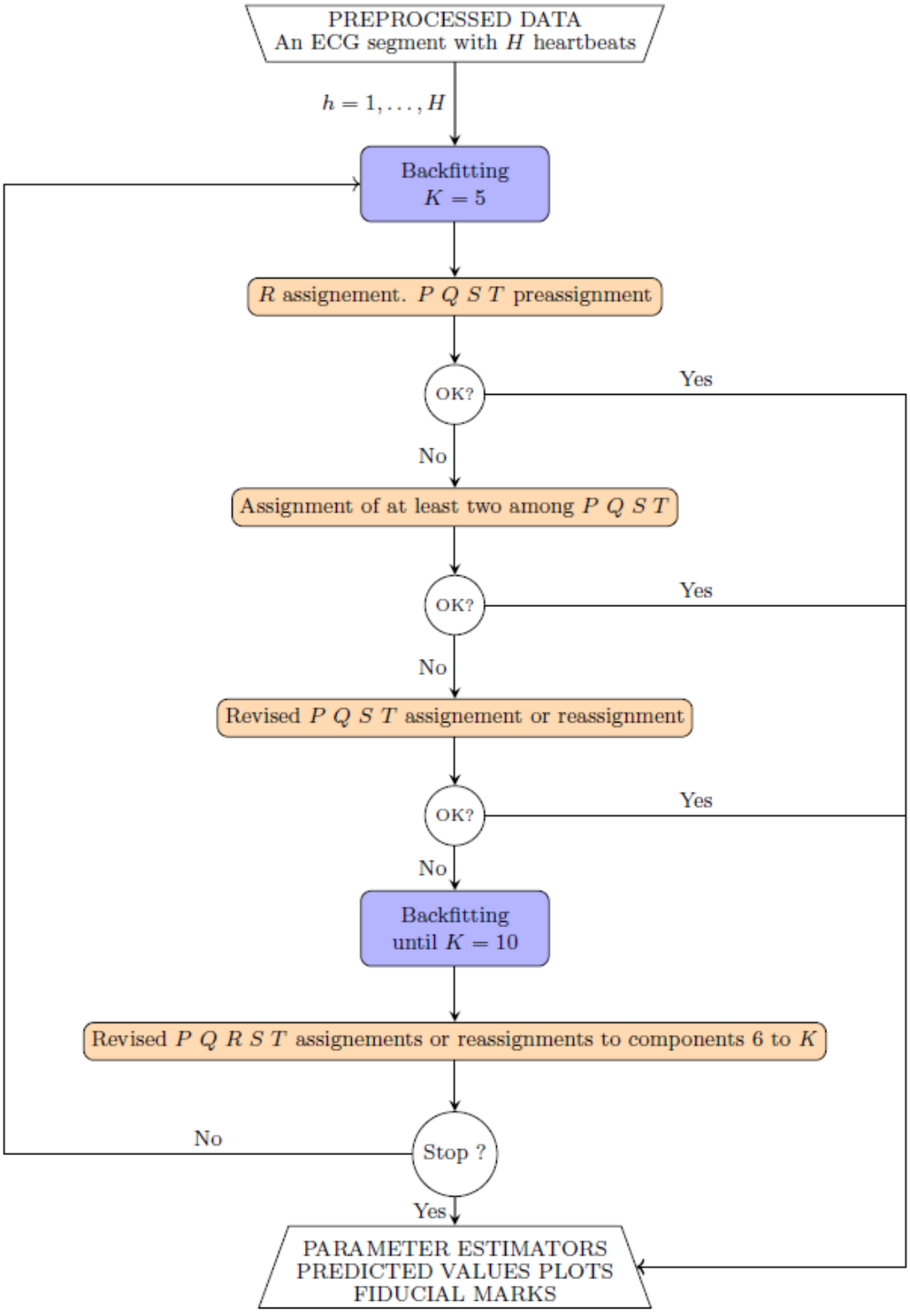


Figura 2.3: Esquema con los pasos llevados a cabo en el ajuste del modelo FMM_{ECG}.

esquemática de los pasos que lo componen. El lector podrá comprobar que el algoritmo es más complejo de lo que puede parecer en un principio, y que justifican las más de 4000 líneas de código empleadas.

A grandes rasgos, los pasos que componen el algoritmo de identificación (paso I) son:

1. Asignación de la onda R.
   a) R es la onda de máxima variabilidad que verifica las restricciones de integridad, entre las que se incluye que sea la que tiene el pico más alto.
   b) Si no se verifican, se relajan un poco las condiciones de integridad, marcando la onda como posible duda.
   c) Se revisa esta asignación si el porcentaje de variabilidad explicada por esta onda es inferior al 10 %. En este caso, se amplía la búsqueda hasta el segundo pico más alto.
   d) En caso de que ninguna verifique las restricciones, elegimos como R la de máxima variabilidad.
2. Preasignación de las primeras cinco ondas.
   a) Se asignan letras a las otras cuatro componentes utilizando las restricciones sobre α. Si todas las ondas muestran una morfología típica de su onda, el algoritmo de identificación **termina**.
   b) En caso contrario, centramos la asignación en la onda de máxima variabilidad (excluyendo la R). Se asigna siguiendo unos thresholds (véase el Axeno A para más detalles) a la letra P, Q, S o T.
   c) Si tras el paso anterior cuadran los parámetros α de las otras tres, se termina. En caso contrario, se utilizan unos thresholds para asignar el máximo número de componentes de entre las 5 ondas encontradas.
   d) Se verifican ciertas restricciones de integridad utilizando, de nuevo, algunos thresholds.
3. Reasignación entre las 5 primeras ondas.
   a) Comprobación de confusiones:
      1) Se comprueba si se ha producido una confusión entre las ondas P y Q (suele ser una situación típica si la morfología es extraña).
      2) Se comprueba si se ha producido una confusión entre las ondas S y T.
      3) Se comprueba si alguna otra onda es mejor candidata para la onda P.
   b) Revisión de las letras asignadas. Se realiza una revisión de todas las ondas que no han sido asignadas (se han encontrado pero el algoritmo no les ha asociado una letra) para ver si alguna resulta mejor que otra ya asignada, por este orden:
      1) Onda T.
      2) Onda P.

3) Onda S.
4) Onda Q.

4. Asignación de letras en componentes hasta la 10. Excepcionalmente alguna onda significativa puede quedar enmascarada por ruido. Se obtienen el ajuste de un modelo FMM_{10} y se repite el paso 3b para las cinco nuevas ondas.

Debido a la enorme cantidad de detalles, clarificaciones, excepciones, thresholds y restricciones de integridad, se ha incluido una versión **muy reducida** de todos los detalles y casuísticas que aparecen en la implementación real.. Por el momento, se prefiere no hacer público un esquema detallado de los pasos que tanto valor añadido aportan al algoritmo, hasta que el artículo en revisión sea finalmente publicado ([3]). Esto quiere decir que no se incluye el código de esta parte en los archivos enviados de forma adjunta a este escrito.

Capítulo 3

Ejemplos de uso

Hasta ahora hemos visto cuatro modelos de forma teórica: el modelo FMM, FMM_m, FMM_m con restricciones y FMM_{ECG}. Sin embargo, no nos hemos centrado en el aspecto práctico de estos modelos. En este capítulo exploraremos algunas de las aplicaciones de estos modelos, incluyendo ejemplos y comparaciones con otros métodos.

En primer lugar, exploraremos algunos modelos sencillos utilizando un modelo FMM con un sólo componente. Incluso en estos casos sencillos, este modelo está dotado de una gran versatilidad que, como veremos, puede ajustarse por ejemplo a diversos patrones de luz emitidos por las estrellas.

A continuación veremos un ejemplo de aplicación de un modelo FMM_m con restricciones en neurobiología. En este caso se observará cómo la actividad cerebral se puede resumir en un modelo oscilatorio donde resulta útil establecer restricciones sobre los parámetros, para el que el FMM produce un ajuste muy bueno. La variedad de disciplinas en las que se puede utilizar es muy grande.

También se proponen una serie de ejemplos utilizando directamente el paquete de software desarrollado para R. En esta sección, dedicada más a los detalles de implementación y del uso del paquete, veremos ejemplos de las distintas funciones que hemos presentado en el Capítulo 1 de este trabajo.

Finalmente, hablaremos de una de las aplicaciones más bonitas del modelo FMM_{ECG} para el ajuste e identificación de las ondas de un ECG. El ajuste en este caso es computacionalmente intensivo, pero su aplicación directa an análisis de grandes cantidades de flujos de datos es inmediata. Analizar una serie de latidos y ofrecer un diagnóstico en tiempo real es, sin duda, un gran avance en el campo de la salud que merece la pena explorar.

3.1. FMM monocomponente: patrones de luz de estrellas

En esta sección exploraremos la versatilidad del modelo FMM con una única componente. Tal y cómo avanzamos en el Capítulo 1, este modelo adecuado siempre que tratemos con señales oscilatorias, presentes por ejemplo en datos sobre expresiones de

genes o patrones luminosos de las estrellas. Todos los análisis han sido realizados utilizando *R*.

Analizaremos datos de luz emitida por estrellas de seis tipos diferentes. Veremos como el modelo FMM es más flexible captando los tipos de intensidad lumínica de los siguientes tipos de estrellas: RR Lyraes (RRab y RRc), Cepheids (Fundamental FU y Overtone FO), Mira y Eclipsing binary (EB).

En la figura 3.1 se muestra una señal típica observada y la señal FMM ajustada para cada uno de los seis grupos de estrellas, excepto para el tipo RRc, puesto que su patrón es muy similar al grupo FO. En su lugar, se muestran dos tipos diferentes para las estrellas EB. Podemos comprobar que el patrón se ajusta razonablemente bien al modelo FMM propuesto. También se muestra el ajuste producido por los modelos Cosinor y FD^2 para su comparación. En la tabla 3.1 se encuentra el porcentaje de variabilidad explicada en cada uno de los ejemplos ajustados, donde el modelo FMM siempre resulta vencedor.

Tipo estrella	Cosinor	FD^2	FMM
RRab	0.526	0.711	0.9
FU	0.758	0.886	0.972
FO	0.852	0.853	0.854
Mira	0.956	0.987	0.992
EB	0.917	0.976	0.978

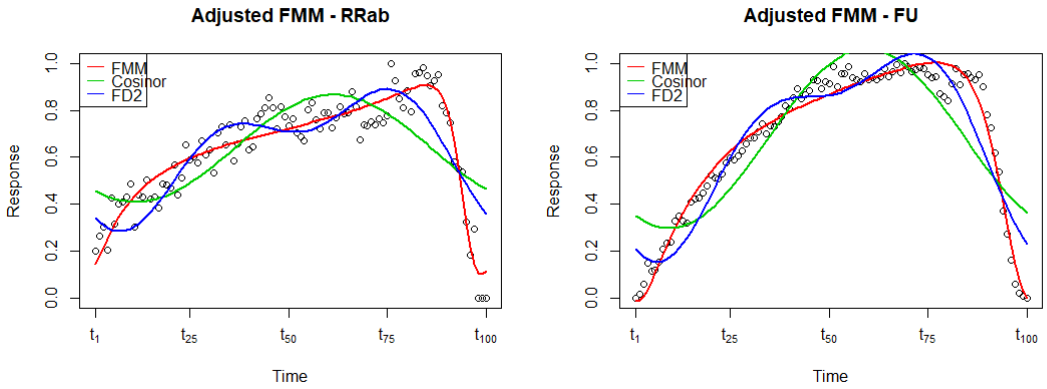
Tabla 3.1: Comparación del estadístico R^2 en los diferentes modelos de ajuste

Una de las grandes ventajas que ofrecen los modelos paramétricos es la posibilidad de incluir una interpretación a sus parámetros. Entre otras cosas permiten, por ejemplo, utilizar modelos de aprendizaje supervisado utilizando estos parámetros como variables, algo que no es posible utilizando un modelo no paramétrico. En [1] se compara la utilidad de estos parámetros, sin entrar en modelos complejos, con otras técnicas empleadas en la literatura, para clasificar las estrellas en un tipo u otro dependiendo de su patrón lumínico. El resultado es que esta técnica es capaz de segmentar en grupos las estrellas mejor que otras muy conocidas en la literatura, como el análisis en componentes principales o la descomposición de Fourier. Puede consultarse el artículo referenciado para más detalles [1].

3.2. Ejemplo de FMM en neurobiología

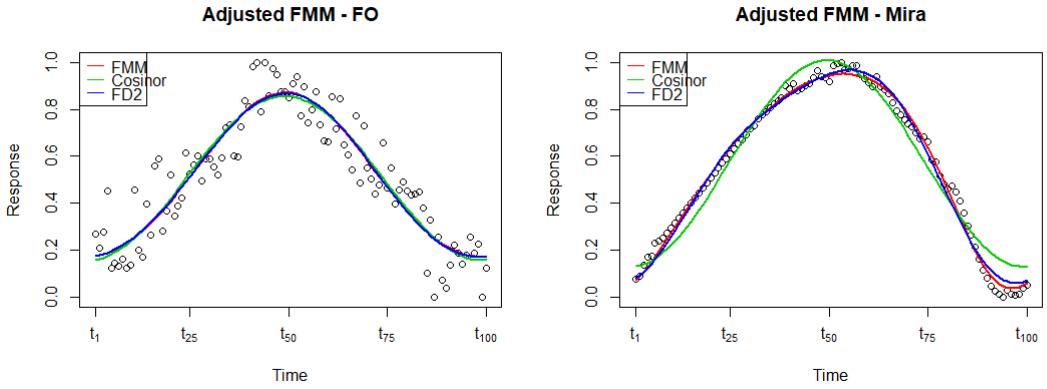
El estudio de la electrofisiología de la actividad neuronal ha sido una de las ramas más estudiadas en neurociencia. Las curvas de potencial de acción pueden considerarse señales oscilatorias. Miden la diferencia de la actividad eléctrica entre las neuronas debido a un estímulo externo. Se ha comprobado que un modelo FMM_2 puede describir esta actividad eléctrica con gran precisión [17].

Habitualmente la neurona responde a un estímulo con múltiples curvas de potencial de acción. Los procedimientos utilizados hasta ahora dividen las diferentes oscilaciones



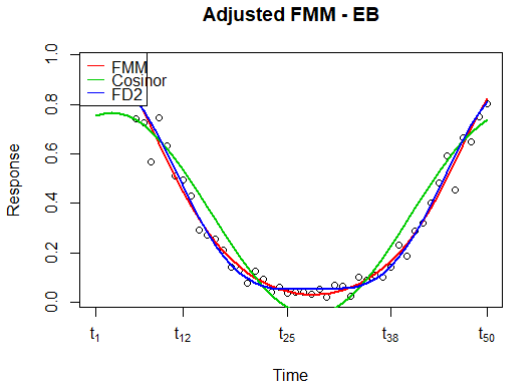
(a) RRab

(b) FU



(c) FO

(d) Mira



(e) EB

Figura 3.1: Patrones de luz emitida por estrellas del tipo RRab (a), FU (b), FO (c), Mira (d), y EB (e), junto con los valores predichos por los modelos FMM, Cosinor y FD2

en sucesivos segmentos y los analizan por separado. Sin embargo, esto puede causar complicaciones, puesto que no todos los segmentos tienen por qué tener la misma longitud. En su lugar, es posible considerar la señal completa y ajustar un único modelo FMM_m. Como cada curva de potencial de acción se ajusta con un FMM_2, si n es el número de segmentos analizamos, debemos tomar $m = 2n$. Es sabido que la forma debe ser la misma en las diferentes curvas, por lo que los parámetros β y ω se pueden fijar para que esto se verifique. El resultado es un ajuste coherente gracias a la interpretabilidad de los parámetros.

En la figura 3.2 se encuentra el ajuste de una señal con tres curvas de potencial de acción. Cada una de ellos tienen el mismo valor de ω, garantizando que los "picos" tienen la misma forma, y el mismo valor para β, garantizando la misma medida de asimetría. En la parte de la izquierda se encuentra el ajuste completo, mientras que en la derecha se muestran las 6 ondas que lo componen por separado.

Podrán encontrarse más detalles sobre este ejemplo en [2].

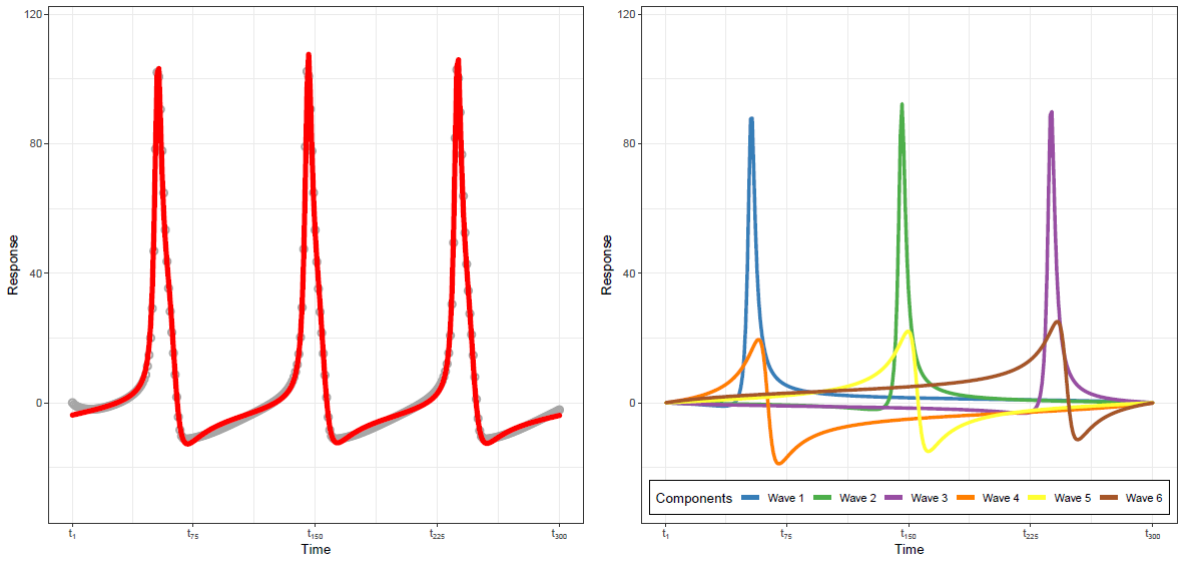


Figura 3.2: Ajuste sobre la actividad electrofisiológica de una neurona.

3.3. Ejemplos de uso del paquete FMM

En esta sección desarrollaremos algunos ejemplos centrándonos en los detalles técnicos del paquete FMM implementado. Es de interés, por tanto, el uso de las distintas funciones, y puede servir a modo de ilustración sobre el potencial del software desarrollado. Incluiremos el código utilizado en los ejemplos, así como sus salidas, para garantizar su reproducibilidad por parte de cualquiera con acceso a las funciones desarrolladas.

Ejemplo 1

En primer lugar generaremos unos datos sencillos sencillos procedentes de un modelo FMM con una componente, más un ruido procedente de una distribución normal. Los parámetros que se utilizan en la simulación son $M = 2$, $A = 1$, $\alpha = 0$, $\beta = 1$, $\omega = 0.1$. Los datos simulados se encuentran en la figura 3.3.

```
example1 <- generate_FMM(2,1,0,1,0.1,sigmaNoise = 0.2,
  outvalues = TRUE)
```

El contenido de *example1* es una lista con dos elementos: un vector de instantes temporales equiespaciados (t), y el vector de datos (y). Por simplicidad, supondremos instantes de tiempo equiespaciados y nos centraremos únicamente en el componente y. Con esto podemos llamar a la función de ajuste *fitFMM*. Como dejaremos el valor de m a 1 (*nback*), internamente se activará la función *fitFMM_ unit*, optimizada para modelos de una componente. Todos los argumentos sobre precisión los dejamos por defecto

```
> fit1.1 <- fitFMM(example1$y)
Time difference of 2.093399 secs
> summary(fit1.1)

Title:
FMM model with 1 components

Coefficients:
M (Intercept): 1.8955
A   alpha    beta   omega
FMM wave 1:   1.0158 6.2003 0.8761 0.0803

Peak and trough times:
t.Upper t.Lower
FMM wave 1:    3.3354   2.9207

Residuals:
Min.      1st Qu.       Median         Mean      3rd Qu.
-0.5163922 -0.1240824 -0.0188065  0.0000008  0.1361186
Max.
0.5298292

R-squared: 0.7833
```

En la figura 3.4 (a) se encuentra una representación gráfica del ajuste junto con los datos simulados. De la información mostrada tras realizar el ajuste extraemos las siguientes conclusiones:

- El ajuste se ha completado en 2.09 segundos.
- Los valores estimados para los parámetros son: $\hat{M} = 1.8955$ ($M = 2$), $\hat{A} = 1.0158$ ($A = 1$), $\hat{\alpha} = 6.2003$ ($\alpha = 0$), $\hat{\beta} = 0.8761$ ($\beta = 1$), $\hat{\omega} = 0.0803$ ($\omega = 0.1$). En todos

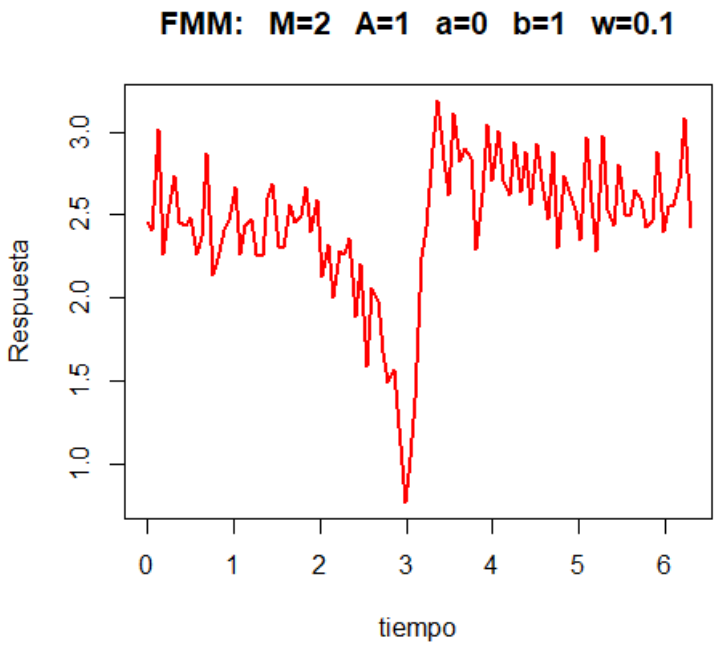


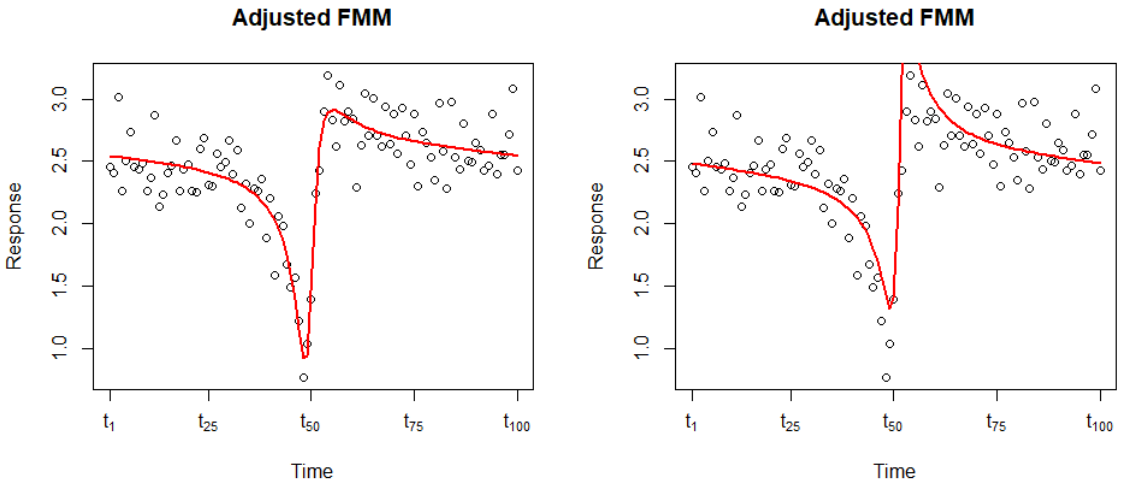
Figura 3.3: Datos simulados para el ejemplo 1 del uso del paquete *FMM*

los casos quedan próximos (recuérdese que hablamos de un orden circular en 2π para el caso de α).

- El instante temporal donde esta onda estimada alcanza su mínimo es $t_L = 2.9207$, mientras que el máximo es $t_U = 3.3354$.
- El valor del R^2, o proporción de variabilidad explicada, es de 0.7833.

Podemos modificar algunos de los parámetros de precisión para tratar de reducir el tiempo de ejecución (*lengthAlphaGrid*, *lengthOmegaGrid* y *numReps*). Realizamos un segundo ajuste obteniendo los siguientes resultados:

```
> fit1.2 <- fitFMM(example1$y,lengthAlphaGrid = 6,
+                  lengthOmegaGrid = 3, numReps = 1)
Time difference of 0.02293706 secs
> summary(fit1.2)

Title:
FMM model with 1 components

Coefficients:
M (Intercept): 2.4649
A   alpha    beta   omega
FMM wave 1:   1.1622 0.0178 1.5595 0.0636
```



(a) Precisión normal (b) Precisión reducida

Figura 3.4: Ajuste del ejemplo 1 ilustrando el uso del paquete *FMM*

```
Peak and trough times:
t.Upper t.Lower
FMM wave 1:   3.2251   2.9709

Residuals:
Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
-0.75618 -0.13587 -0.02575 -0.02749  0.11738  0.58946

R-squared: 0.6149
```

En este caso el tiempo de ejecución se ha reducido a apenas 0.02 segundos, aunque a costa de reducir la proporción de variabilidad explicada a 0.6149. En la figura 3.4 (b) se muestra este ajuste, claramente peor que el anterior.

Puede compropbarse que con los siguientes parámetros de precisión:

```
fit1.3 <- fitFMM(example1$y,lengthAlphaGrid = 6,
lengthOmegaGrid = 3, numReps = 2)
Time difference of 0.04487991 secs
```

Es decir, tan sólo una repetición más que el anterior, se obtiene una precisión equivalente al primer ajuste con tan sólo 0.0449 segundos, a diferencia de los más de 2 segundos necesarios con los valores por defecto. La precisión necesaria para alcanzar buenos resultados depende en gran medida del problema, pero por defecto se mantienen unos valores lo suficientemente grandes como para garantizar que incluso un usuario sin mucho conocimiento del software obtenga resultados aceptables.

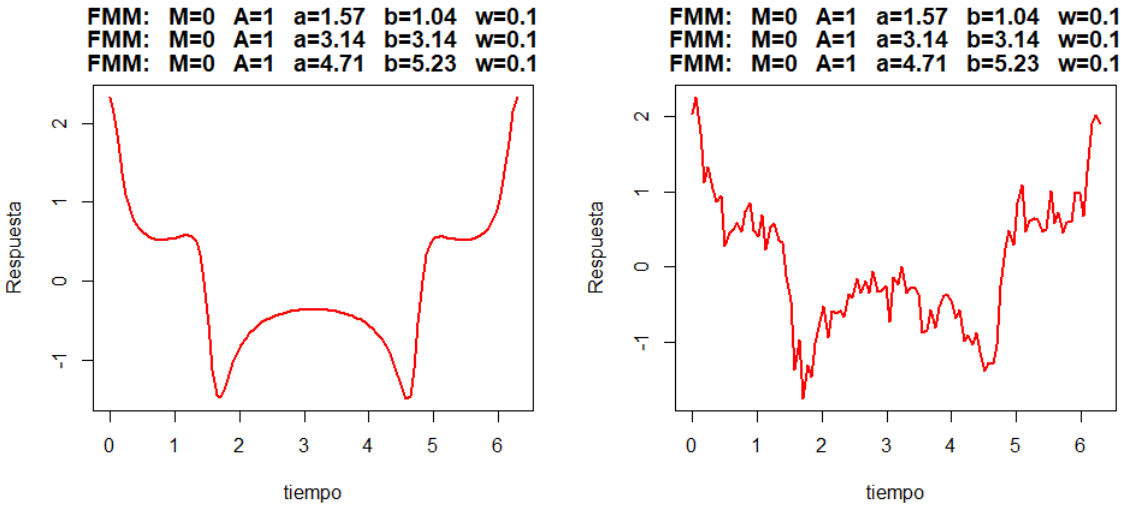
Ejemplo 2

En este ejemplo simularemos un modelo FMM_3 con el añadido ruido gaussiano que hemos venido incluyendo hasta ahora. De forma similar, realizaremos diferentes ajustes e ilustraremos el funcionamiento de la función de ajuste, incluyendo algunos de los parámetros que conciernen al ajuste múltiple de *backfitting*. Tal y como se refleja en la sección 1.4, la función interfaz redigirá la llamada a *fitFMM_back* o a *fitFMM_restr* según convenga.

Los parámetros de las tres ondas son los siguientes:

$$
\begin{aligned}
&M = 0 \\
&A_1 = A_2 = A_3 = 1 \\
&\alpha_1 = \pi/2, \alpha_2 = \pi, \alpha_3 = 3\pi/2 \\
&\beta_1 = \pi/3, \beta_2 = \pi, \beta_3 = 5\pi/3 \\
&\omega_1 = \omega_2 = \omega_3 = 0.1
\end{aligned}
$$

A esta señal añadimos un ruido gaussiano con desviación estándar $\sigma = 0.2$. En la figura 3.5 se pueden ver los datos generados.



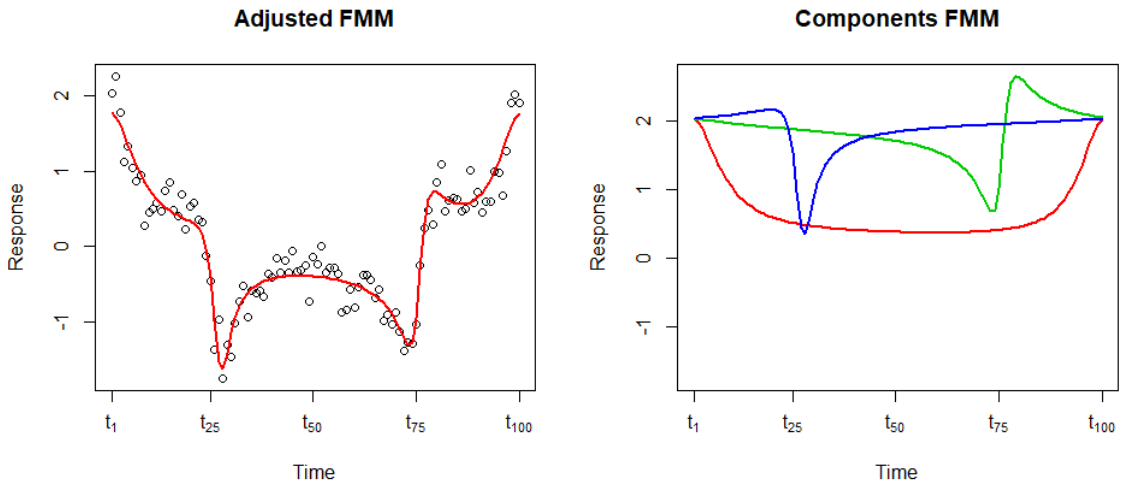
(a) Patrón subyacente (b) Datos generados

Figura 3.5: Patrón subyacente y datos generados para el ejemplo 2

En primer lugar realizamos un ajuste con todos los parámetros por defecto, especificando que debe buscar un modelo con 3 componentes. El resultado del ajuste se encuentra en la figura 3.6. En el panel (a) se muestra el gráfico con el modelo ajustado, y en el panel (b) se presentan las diferentes ondas.

```
> fit2.1 <- fitFMM(example2$y,nback = 3)
```

```
|----------------------------------------------------|
|====================================================|
Stopped by reaching maximum iterations
Time difference of 18.78574 secs
> summary(fit2.1)

Title:
FMM model with 3 components

Coefficients:
M (Intercept): -0.221
A   alpha    beta   omega
FMM wave 1:   0.8346 3.0905 2.9964 0.2254
FMM wave 2:   0.9928 1.5510 1.3504 0.0883
FMM wave 3:   0.9199 4.7587 5.5922 0.0792

Peak and trough times:
t.Upper t.Lower
FMM wave 1:    6.2020   3.6517
FMM wave 2:    4.8493   4.4886
FMM wave 3:    1.1212   1.6113

Residuals:
Min.    1st Qu.     Median       Mean    3rd Qu.       Max.
-0.608217 -0.147371 -0.006785  0.000000  0.144203  0.561609

R-squared: 0.9324
```



(a) Modelo ajustado (b) Componentes ajustadas

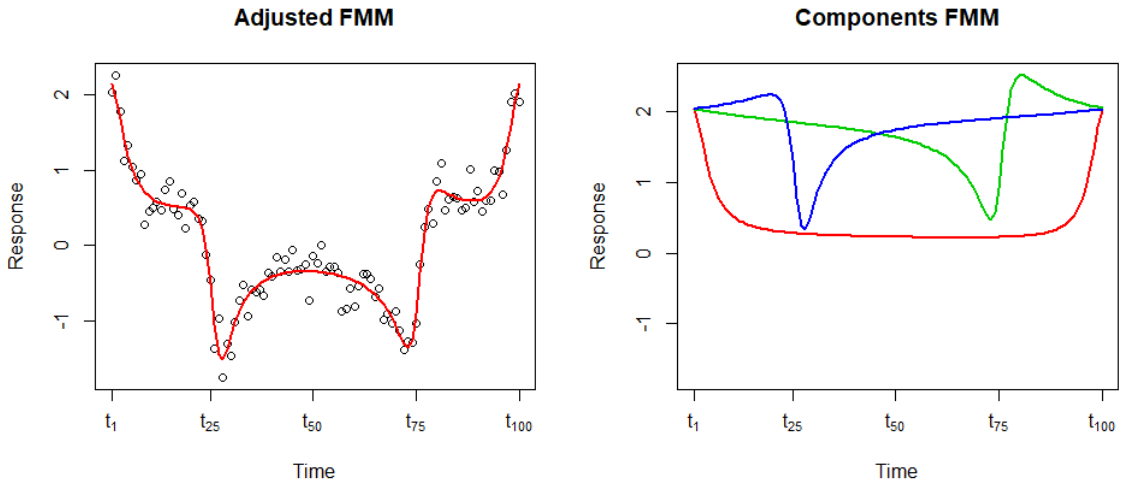
Figura 3.6: Primer ajuste realizado para el ejemplo 2 ($R^2 = 0.9324$)

Algunos datos interesantes sobre este ajuste son los siguientes:

- El tiempo necesario para llevar a cabo el ajuste es de 18.78 segundos. Se realizan tres iteraciones de *backfitting*. Podría reducirse el tiempo de ejecución reduciendo la precisión, de forma similar a lo mostrado en el ejemplo anterior.
- Los parámetros estimados están cerca de los reales, a excepción de $\hat{\omega_1} = 0.2254$, que se aleja del valor real $\omega = 0.1$.
- La proporción de variabilidad explicada es de 0.9324.

No existe un criterio de parada por defecto, de forma que la función termina al realizar su número máximo de iteraciones (3 en este caso). En el siguiente ajuste, especificamos que se utilice un criterio basado en el R^2 como criterio de parada. Mientras haya una mejora de al menos un 0.1 % con respecto al anterior ajuste, se mantiene la búsqueda. Se coloca a 10 el número máximo de iteraciones. El resultado final es que se llevan a cabo 7. En la figura 3.7 se encuentran la representación gráfica de este ajuste.

```
> fit2.2 <- fitFMM(example2$y,nback = 3, maxiter = 10,
+                  stopFunction = R2(difMax = 0.001))
|-------------------------------------------------|
|=================================================|
Stopped by the stopFunction (7 iterations)
Time difference of 43.84409 secs
> summary(fit2.2)

Title:
FMM model with 3 components

Coefficients:
M (Intercept): -0.0391
A  alpha   beta  omega
FMM wave 1:  0.9197 3.0995 3.0145 0.1213
FMM wave 2:  1.0162 1.5358 1.2188 0.1097
FMM wave 3:  0.9582 4.7284 5.4298 0.0989

Peak and trough times:
t.Upper t.Lower
FMM wave 1:   6.1937  4.0029
FMM wave 2:   4.9262  4.4618
FMM wave 3:   1.0955  1.6138

Residuals:
Min.    1st Qu.     Median       Mean    3rd Qu.       Max.
-0.464524 -0.110112  0.008482  0.000000  0.133752  0.472640

R-squared: 0.9507
```



(a) Modelo ajustado

(b) Componentes ajustadas

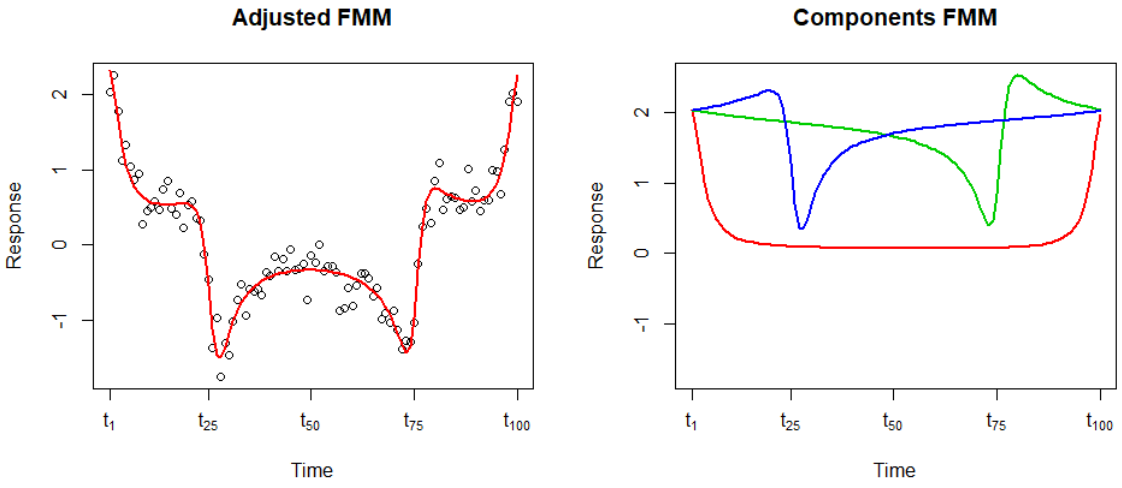
Figura 3.7: Segundo ajuste realizado para el ejemplo 2 ($R^2 = 0.9507$)

El tiempo de ejecución se multiplica por tres, llegando hasta los 43.84 segundos. A cambio, obtenemos un valor de R^2 de 0.9507 y los estimadores para ω ya se acercan a la realidad: $\hat{\omega_1} = 0.1213$, $\hat{\omega_2} = 0.1097$ y $\hat{\omega_1} = 0.0989$.

Finalmente, se propone añadir restricciones sobre el parámetro ω, para que todas las ondas verifiquen una restricción de igualdad $\omega_1 = \omega_2 = \omega_3$. Esto hará disminuir parcialmente el valor del R^2, puesto que estamos imponiendo una limitación al ajuste que maximiza este valor. Sin embargo, es preferible cuando tenemos un cierto conocimiento del dominio del problema. Además, para agilizar el proceso de búsqueda, activamos la paralelización. El resultado gráfico se encuentra en la figura 3.8.

```
> fit2.3 <- fitFMM(example2$y,nback = 3, maxiter = 10,
+                  stopFunction = R2(difMax = 0.001),
+                  omegaRestrictions = c(1,1,1),parallelize = TRUE)
showProgress not available when specifying omegaRestrictions
Time difference of 7.959825 secs
> summary(fit2.3)

Title:
FMM model with 3 components

Coefficients:
M (Intercept): 0.0653
A  alpha   beta  omega
FMM wave 1:  0.9960 3.1169 3.0944 0.0979
FMM wave 2:  1.0713 1.5317 1.1907 0.0979
FMM wave 3:  0.9957 4.7075 5.3234 0.0979
```

```
Peak and trough times:
t.Upper t.Lower
FMM wave 1:    6.2003   3.5267
FMM wave 2:    4.8976   4.4779
FMM wave 3:    1.1311   1.6049

Residuals:
Min.    1st Qu.     Median       Mean    3rd Qu.       Max.
-0.489962 -0.098398  0.009212  0.000000  0.119831  0.492208

R-squared: 0.9476
```



(a) Modelo ajustado

(b) Componentes ajustadas

Figura 3.8: Tercer ajuste realizado para el ejemplo 2 ($R^2 = 0.9476$)

Con este ajuste se permite que el estimador para ω sea siempre el mismo, $\hat{\omega} = 0.0979$, muy próximo al valor real de 0.1, y en un total de apenas 8 segundos. El valor de $R^2 = 0.9476$ es muy próximo al del anterior ajuste.

Por supuesto, quedan más ejemplos de ajuste que se podrían probar. Algunas de los aspectos más importantes que no hemos tratado aquí son:

- Restricciones sobre el parámetro β.
- Ajuste para varios períodos.
- Predicción para nuevos valores temporales.
- Funcionalidad gráfica del paquete *ggplot2*.

Sin embargo, no es el objetivo de este trabajo el de sobrecargar con ejemplos que ilustren al detalle toda la funcionalidad incluida. Si el lector desea conocer más información, puede dirigirse al repositorio oficial de R (CRAN) y descargar el paquete junto con la ayuda incluida. Está previsto que la versión final se publique antes de finales de 2020 [2], [9].

3.4. Ejemplos de ECG

La base de datos de QT, públicamente disponible en www.physionet.org, ha sido escogida para llevar a cabo los ejemplos del modelo FMM_{ECG} porque es un repositorio de datos muy utilizado por diversos autores que proponen análisis para ECGs. Posee una amplia variedad de morfologías de ECG, algunas de ellas asociadas a ciertas patologías. En total, la base de datos contiene 105 ECG con las señales medidas desde dos derivaciones. En nuestro caso, sólo nos centraremos en la derivación 2, la más famosa. En total se han analizado 3623 latidos.

El siguiente ejemplo de funcionamiento del modelo FMM_{ECG} ha sido extraído de [3].

Para cada uno de estos latidos se lleva a cabo el ajuste del modelo FMM_{ECG}. El coeficiente de determinación, que mide la proporción de variabilidad explicada, es superior a 0.98 en término medio.

Se han elegido seis categorías para ilustrar el funcionamiento del ajuste del modelo:

- NORMAL: patrón típico.
- PACE: latido con ritmo acelerado.
- RBBB: bloqueo de rama derecha.
- APC: latido auricular prematuro.
- PVC: contracción ventricular prematura.
- NOISY: patrón con ruido.

En la figura 3.9 se encuentra un patrón típico para cada una de estas ondas, junto con su ajuste. También se incluye la detección automática de las ondas P, R y T, llevada a cabo gracias a la interpretabilidad del modelo. En todos los casos, el valor de R^2 es superior a 0.98, excepto para el caso más ruidoso, donde se mantiene cerca de 0.92. Resulta evidente que el ajuste es muy bueno, empleando tan sólo 20 parámetros.

La identificación automática de las ondas P y T resulta evidente dada la formulación del modelo. Cada onda se corresponde con una señal oscilatoria ajustada, por lo que basta con encontrar el instante temporal del pico donde se produce. Este problema aún plantea problemas incluso en las propuestas más recientes. En el modelo FMM_{ECG}, sea t_J^{FI}, $J = P, T$ el *fiducial point* que marca la localización de la onda. Dependiendo de los parámetros del modelo, $t_J^{FI} = t_J^U$ o $t_J^{FI} = t_J^L$.

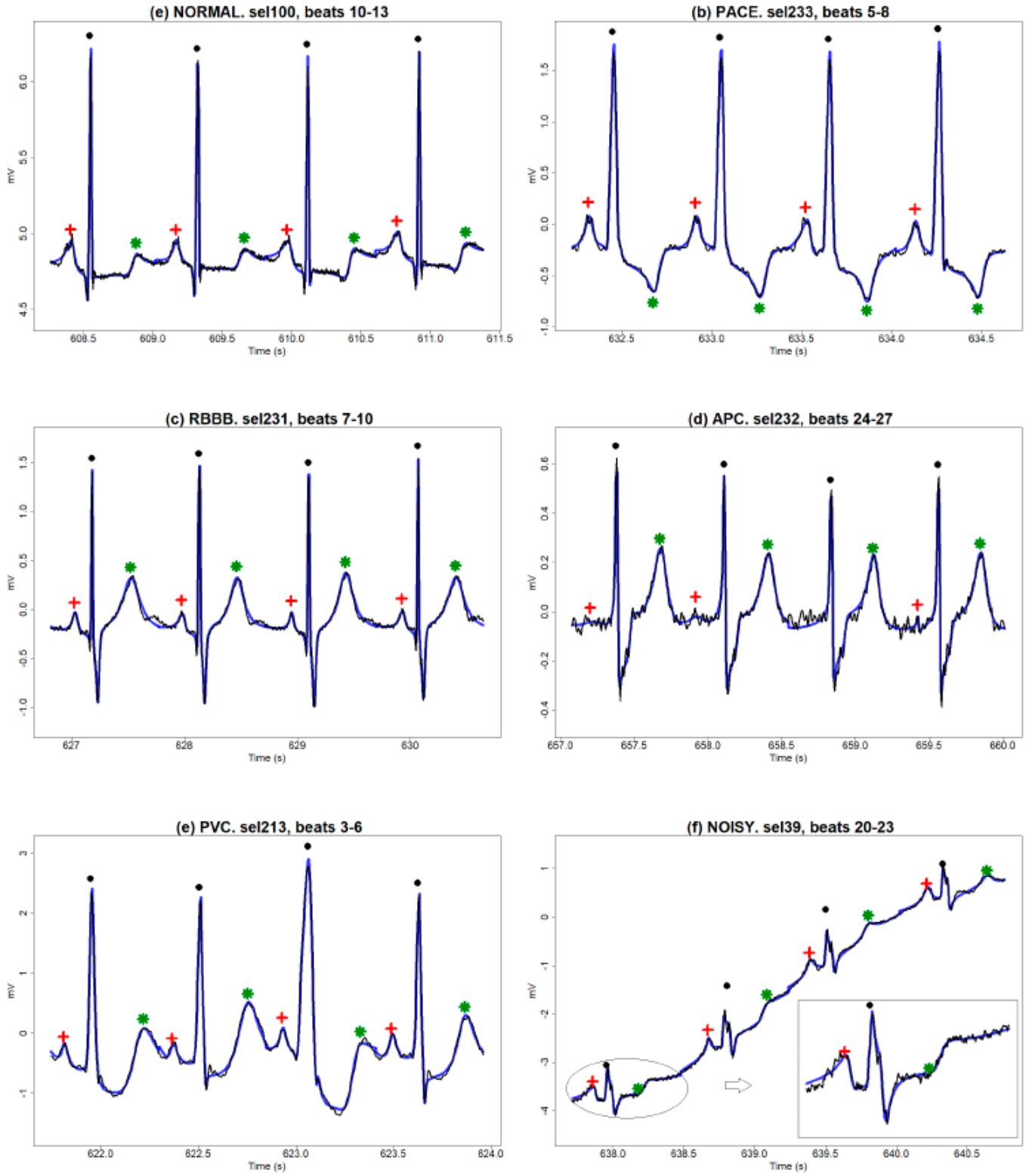


Figura 3.9: Segmentos ECG en negro, junto con su correspondiente ajuste en azul, y marcas sobre las rondas P (rojo), R (negro) y T (verde).

En la ruta relativa *FMMECG/QT* se pueden encontrar los ajustes individuales para algunos de los individuos, además del código que permite generarlos. No se incluyen todos porque suponen un volumen de datos muy elevado para ser parte del adjunto de este trabajo.

Como último ejemplo, se adjunta un par de latidos que pueden encontrarse en dicha ruta. El primero es del individuo 114 (QT1, latido 10) (figura 3.10). Podemos ver que la onda P es muy débil si la comparamos con un ECG típico. No obstante, el programa reconoce todas las ondas y el ajuste final es satisfactorio. Un ejemplo no tan bueno viene de la mano del individuo 39 (QT2, latido 20) (3.11). En este caso, aparece una onda marcada como Q cuando en realidad se trata de una P, y se marca como P algo que en realidad no es más que ruido. A pesar de esta confusión, el ajuste numérico es bueno incluso en presencia de ruido.

La gran ventaja de la utilización del modelo FMM_{ECG} en el análisis de ECG es que proporciona una serie de parámetros que lo hacen interpretable. Caracterizamos cada una de las cinco ondas de forma única, y funciona muy bien tanto en ECG típicos como en aquellos con ruido o diferentes preprocesamientos.

Estos parámetros pueden resultar de utilidad en la diagnosis automática de enfermedades. Aunque en este trabajo no hemos entrado en cómo podrían utilizarse, existen dos principales aproximaciones. Una de ellas parte de utilizar los parámetros como entradas de algún algoritmo de aprendizaje automático. Por otro lado, también se puede buscar una interpretación más profunda a los parámetros. Por ejemplo, si la contracción ventricular prematura sucede cuando la repolarización ventricular sucede muy rápidamente, esto se traduce en que la onda S y la T van a aparecer mezcladas. Si conseguimos identificarlas, y la distancia entre ellas es menor de la habitual, podemos tener un sistema automático de diagnóstico para este caso. Por supuesto, este caso particular no es más que una aproximación simplista, pero la idea general que subyace es la misma.

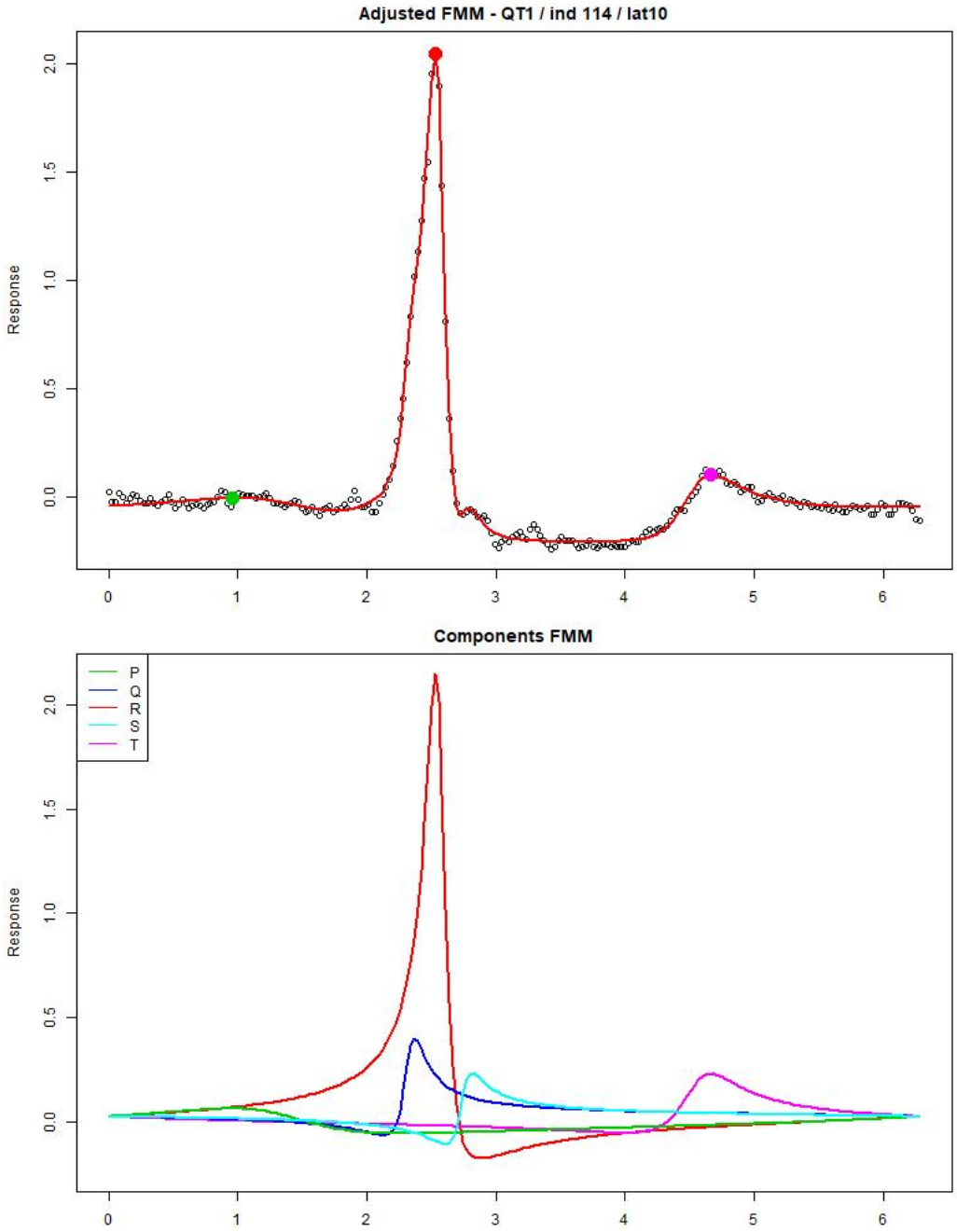


Figura 3.10: Individuo 114 (QT1), latido 10

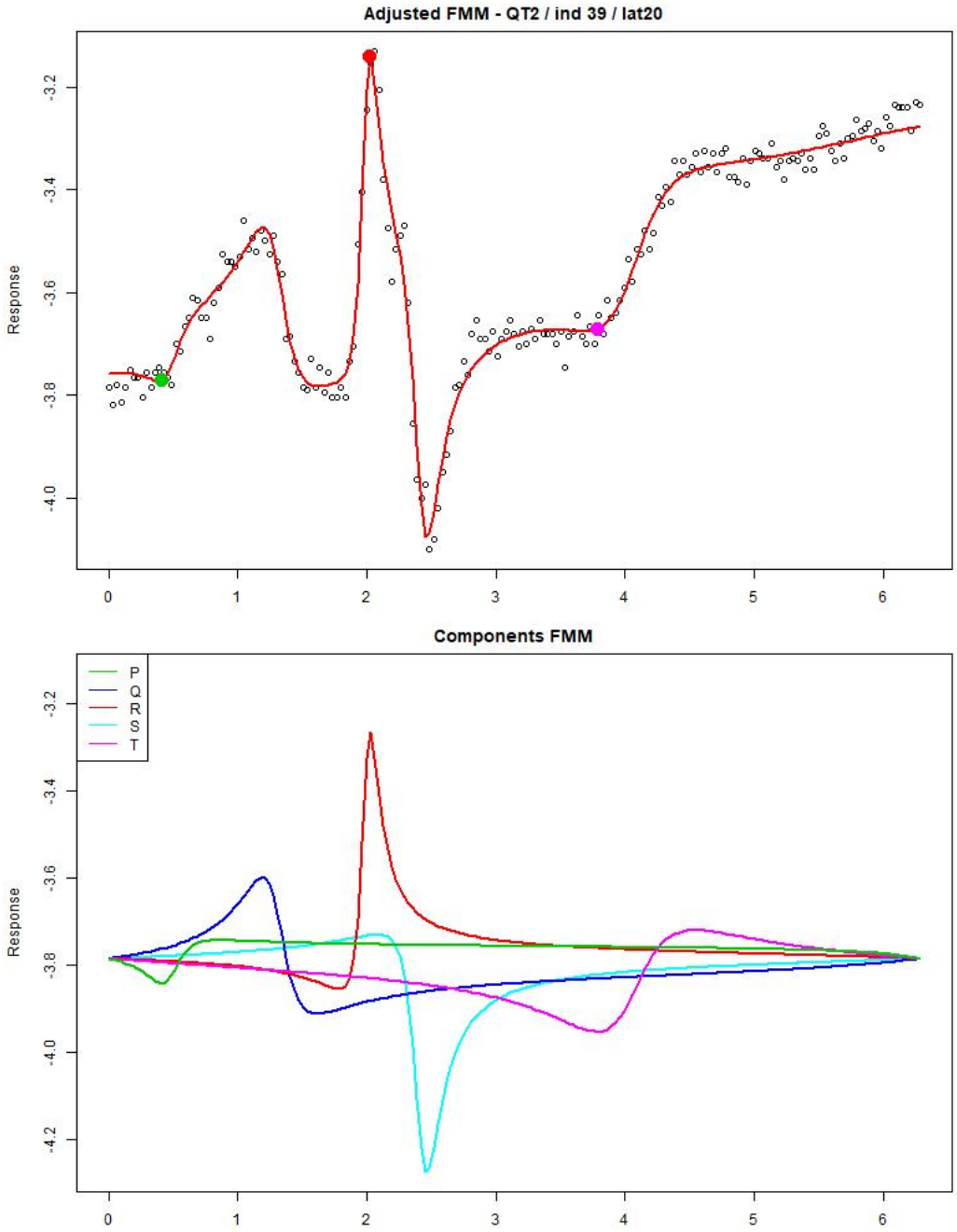


Figura 3.11: Individuo 39 (QT2), latido 20

Conclusiones y trabajo futuro

Como hemos podido comprobar en este trabajo, los patrones rítmicos y oscilatorios aparecen de forma natural en muchas aplicaciones, entre las que se incluyen biología, astrofísica, neurobiología y electrocardiografía. Hasta ahora, los modelos más utilizados son el modelo Cosinor y de descomposición de Fourier, pero que tienen importantes limitaciones. El primero falla a la hora de describir morfologías amplias en muchos sistemas, puesto que es un modelo demasiado simple y sólo capaz de describir patrones simétricos. El modelo de descomposición de Fourier es más versátil (de hecho, es una generalización del Cosinor), pero en muchos casos se necesitan muchos parámetros para formas complejas, lo que conduce irremediablemente a problemas de sobreajuste.

En este trabajo se ha expuesto el modelo FMM junto a sus variantes FMM_m y FMM_m restringido, desarrollado por el grupo de investigación, que es capaz de adaptarse a una gran variedad de patrones oscilatorios utilizando pocos parámetros, y se ha realizado una implementación del mismo. De hecho, la forma de cada onda se describe de forma precisa con cuatro parámetros que representan respectivamente la amplitud, localización, asimetría y apuntamiento. Se adapta de forma natural a muchos ejemplos, como el patrón de luz emitido por las estrellas, la actividad eléctrica de las neuronas, o el ajuste de electrocardiogramas. Además, es un modelo paramétrico, lo que sirve para caracterizar las ondas y definir algunas características interesantes, como el instante temporal de máxima expresión de un gen o el momento de máxima emisión de una estrella.

Por tanto, la contribución más importante de este trabajo es la implementación de un modelo computacional capaz de adaptarse a muchas situaciones. Este trabajo realizado es el germen de un artículo para la revista *Journal of statistical software*, previsto para finales de 2020, junto con la publicación en el CRAN oficial de R del paquete que se adjunta a este trabajo [2].

Existen algunas mejoras previstas a corto plazo para este paquete, que se sucederán como actualizaciones de versión. Entre ellas, se encuentra una mejora en la eficiencia en la estimación para el caso de restricciones sobre el parámetro ω. Tal y como se detalló en la sección de implementación, es necesario extraer el *grid* de este parámetro fuera del algoritmo de estimación y repetirlo múltiples veces, con la correspondiente pérdida de rendimiento. La paralelización es una alternativa en la que estamos trabajando. Las dos aproximaciones que se han planteado de forma teórica se pretenden evaluar y añadir como una actualización a la versión desarrollada.

Por otro lado, la siguiente gran contribución es la implementación del algoritmo para

el ajuste de un ECG. Es bien sabido que este tema siempre ha suscitado mucho interés en la comunidad científica, por ser un reto que cuenta con pocas soluciones válidas, así como de máximo interés sanitario. Los expertos cardiólogos necesitan revisar de forma manual las diferentes representaciones y se basan en su experiencia para un diagnóstico. Hasta el momento, no existe un método automático con una precisión superior a la inspección humana, por lo que la cantidad de recursos que se necesita es enorme.

La aproximación propuesta por el modelo FMM_{ECG} dota de una interpretabilidad al modelo de la que podemos presumir. Muchas técnicas de aprendizaje automático, además de no tener una buena precisión, actúan como caja negra. ¿Cómo se puede convencer a un cardiólogo de que la interpretación automática es mejor que su ojo, si no podemos dar argumentos a favor? Este gran problema se solventa con el modelo que proponemos, avalado por un artículo actualmente en proceso de revisión [3].

El siguiente gran reto en este campo se divide en dos partes. Por un lado, gracias a los parámetros del modelo, podemos estudiar las diferencias entre un ECG "corriente" y alguno con ciertas patologías. Viendo sus diferencias gráficamente y cómo impacta sobre los parámetros, podemos ser capaces de dar un diagnóstico automático. A corto plazo, está previsto estudiar el impacto del infarto de miocardio.

Por último, pero no menos importante, es su aplicación al análisis de grandes flujos de datos. Un electrocardiógrafo es una máquina que captura la actividad eléctrica del corazón. Actualmente se necesitan entre 30 y 60 segundos para el análisis de un único latido. El siguiente paso pasa por optimizar el procedimiento, siendo capaces de capturar datos en tiempo real y producir una interpretación (e incluso un diagnóstico) automática. Actualmente se necesita esperar a que los resultados sean analizados de forma manual, y, en algunas patologías, como el infarto de miocardio, se necesita una atención inmediata.

Además, muchos dispositivos electrónicos se están empezando a conectar a la red para enviar datos, como los modelos más novedosos de marcapasos. Estos dispositivos registran la actividad y la envían a un servidor, donde un cardiólogo puede revisar si hay alguna anomalía. Ni que decir tiene que el volumen de datos generado por este procedimiento es enorme. Una técnica de interpretación y diagnosis automática en tiempo real podría ser muy efectiva. Por ejemplo, se podría diseñar un sistema de alertas que envíe una notificación cuando algo se salga de lo común.

Con todo, hemos visto que el modelo FMM, aunque muy versátil, tiene limitaciones. En muchos casos es necesario desarrollar una metodología que, aunque basada en el FMM, tiene rasgos específicos del dominio (como el ejemplo de los ECG). Además, parte de ciertas suposiciones que no siempre son ciertas: ruido Normal, la existencia de un mínimo local por cada máximo, etc. Sin embargo, no debemos olvidar que esto pasa con todos los modelos paramétricos desarrollados.

Como se suele decir en una famosa cita atribuida a George Box, "en esencia , todos los modelos están equivocados, pero algunos son útiles".

Bibliografía

[1] Rueda, C., Larriba, Y. and Peddada, S. (2019). A Frequency Modulated Mobius Model Accurately Predicts Rhythmic Signals in Biological and Physical Sciences. Scientific Reports, 9, 1. pag 1-10

[2] (PREPRINT) Lamela A., Fernández I., Larriba Y., Rodríguez-Collado A., and Rueda C. (2020). FMM: An R package for modeling rythmic patterns in oscillatory systems.

[3] (PREPRINT) Rueda, C., Larriba, Y., and Lamela, A. (2020). The hidden waves in the ECG uncovered: A multicomponent model for the Cardiac Rhythm. arXiv preprint arXiv:2005.10173.

[4] Rueda, C., Rodríguez-Collado, A., and Larriba, Y. (2020). A novel wave decomposition for oscillatory signals. arXiv e-prints, arXiv-2006.

[5] G. Cornelissen. (2014). Cosinor-based rhythmometry. Theoretical Biology and Medical Mod-elling. 11(1):16, 2014. doi: 10.1186/1742-4682-11-16.

[6] Lamela A. Fernández M.A. (2019). Análisis y mejoras en algoritmos de cálculo de estimadores bajo restricciones Up-Down-Up y su aplicación en cronobiología. Trabajo fin de Grado en Estadística, Universidad de Valladolid.

[7] Larriba, Y., Rueda, C., Fernández, M.A. and Peddada, S. (2020) Order Restricted Inference in Cronobiology. Statistics In Medicine. vol. 39, no 3, p. 265-278.

[8] J.A. Nelder and R. Mead. (1965). A simplex method for function minimization. The Com-puter Journal, 7(4):308 313, 1965. doi: 10.1093/comjnl/7.4.308.

[9] R Core Team (2018). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

[10] Bayés de Luna A. (2014). Electrocardiografía básica. Grupo Menarini. ISBN 978-84-936320-8-3.

[11] McSarry P., Cli ord G., Tarassenko G., Smith L. (2003). A dynamical model for generating synthetic electrocardiogram signals. IEEE transactions on biomedical engineering 50, 289-294.

[12] Roopa, A. & Harish, B. F. (2017). A survey on various machine learning approaches for ecg analysis. International Journal of Computer Applications 163, 25-33.

[13] P lawiak, P. Novel. (2018). Methodology of cardiac health recognition based on ecg signals and evolutionary-neural system. Journal of The Royal Society Interface 92, 334-349

[14] Pan, J. & Tompkins, W. (1985). A real-time qrs detection algorithm. IEEE Trans. Biomed. Eng 32, 230-236.

[15] Friganovic, K., Kukolja, D., Jovic, A., Cifrek, M. & Krstacic, G. (2018). Optimizing the detection of characteristic waves in ecg based on processing methods combinations. IEEE access 6, 50609-50626.

[16] En Wikipedia. Mean of circular quantities. Recuperado el 18 de octubre de 2020 de https://en.wikipedia.org/wiki/Mean_of_circular_quantities.

[17] En Wikipedia. Potencial de Acción. Recuperado el 24 de noviembre de 2020 de https://es.wikipedia.org/wiki/Potencial_de_acción.

Índice de figuras