# DETECCIÓN DEL COMPLEJO QRS EN UN ELECTROCARDIOGRAMA (ECG): IMPLEMENTACIÓN DEL ALGORITMO PAN-TOMPKINS.

**Prac4.m**: En este fichero realizaremos diversas tareas para implementar la detección del complejo QRS mediante al algoritmo Pan-Tompkins a partir de algunas señales de electrocardiogramas (ECG) proporcionadas. De manera resumida, el proceso consistirá en cargar la señal del ECG a procesar, centrarla eliminando su "offtset" como ya se hizo en la práctica 2, y representarlo gráficamente tanto en el tiempo como en frecuencia. A partir de ahí se aplicará el algoritmo indicado, realizando dos filtrados (uno pasa bajos y otro pasa altos) para reducir las frecuencias que no estén entre 5 y 15 Hz (las principales del complejo QRS), se derivará la señal aplicando un nuevo filtro que aproxime dicha operación, se elevará al cuadrado toda la señal y se aplicará una ventana de integración de 150 ms, dejando una señal que presentará ciertas "acumulaciones" de valores que podrán corresponder o no a complejos QRS, para lo cual habrá que aplicar un algoritmo de decisión, del cual se podrán hacer diversos enfoques dependiendo de lo complejo que queramos hacerlo, habiendo varias mejoras que se pueden aplicar de forma incremental al algoritmo inicial. El resultado que ofrecerá la ejecución del script de MATLAB será una gráfica que, superpuesta al ECG integrado y junto con el ECG original, nos marcará los puntos donde se haya detectado un complejo QRS y la evolución de los umbrales de detección.

El alumno tendrá que ir rellenando los apartados que se indican en el presente guion y al finalizar la práctica deberá subir a la tarea de la Práctica 4 del Campus Virtual un fichero comprimido con el archivo de MATLAB completado y las respuestas a las preguntas que se realizan (en formato Word o PDF). El archivo de MATLAB deberá comentarse para indicar qué tarea se está realizando en todo momento, y si se generan figuras "extra" para realizar comparaciones de gráficas, por ejemplo, éstas se comentarán también en la memoria que se entregue adjuntando alguna captura de las mismas para apoyar la explicación.

## 1.- Detección del complejo QRS.

Se proporciona un script de MATLAB relleno parcialmente y que el alumno deberá completar para realizar el algoritmo Pan-Tompkins, el cual se encarga de detectar el complejo QRS en una señal de un electrocardiograma (ECG), y que posee diversas fases, tal y como se estudió en teoría. Los pasos a seguir para la creación del script de MATLAB están resumidos a continuación:

1. Cargar la señal del ECG de un fichero externo, para lo cual se proporcionan en el Campus Virtual 13 señales distintas para que se aplique el script de MATLAB sobre ellas (el script sólo debe procesar 1 de ellas en cada momento). Dejaremos configurada la carga de todas las señales, pero el "ECG" a tratar en el script será solamente uno, y para cambiar de señal se modificará al principio del script la asignación de la variable "ECG" al fichero cargado, dejando el resto como comentarios. Por ejemplo:

```
load ('E1000000.MAT');
load ('E1040552.MAT');
load ('E1001103.MAT');
...
ECG_1 = e1000000;
% ECG_1 = e1040552;
% ECG_1 = e1001103;
...
```

2. Figura 1. Eliminar el OFFSET de esta señal del ECG, como se vio en la práctica 2, y representar gráficamente la señal en función del tiempo como el primer "subplot" (2x1) de la Figura 1, sabiendo que la frecuencia de muestreo de la señal es de 500 Hz.

3. Realizar un primer análisis en frecuencia (FFT) de toda la señal del ECG sin OFFSET y representar gráficamente la magnitud en función de la frecuencia (en Hz) en un segundo "subplot" (2x1) dentro de la misma Figura 1, dibujando sólo las frecuencias entre 0 Hz y Fs/2 Hz.

4. Figura 2. Representar de nuevo la señal del ECG sin OFFTSET en función del tiempo como el primer "subplot" (2x1) de la Figura 2, y en el segundo subplot la magnitud de la FFT, pero centrándose en este caso sólo en las frecuencias entre 0 y 35 Hz. El resto de representaciones en frecuencia de la práctica deberán ser también entre 0 y 35 Hz únicamente, para centrarnos en la parte importante de nuestra señal.

5. Figura 3. Aplicar un filtro pasa-bajos a la señal del ECG (H1), para lo cual se ha creado una adaptación de la función de transferencia H(z) del filtro original del algoritmo de Pan-Tompkins de manera que sirva con la nueva frecuencia de muestreo de 500 Hz (la original era de 200 Hz). Dicho filtro deberá ser introducido en MATLAB, normalizado, y aplicado sobre la señal del ECG a la que le habíamos calculado la Transformada de Fourier (FFT) previamente. Se representará gráficamente esta nueva señal en frecuencia tras aplicar el filtro en el segundo "subplot" (2x1) de la Figura 3, y tras calcular la IFFT dibujaremos en el primer "subplot" (2x1) la señal del ECG en función del tiempo una vez filtrada con H1 (esta FFT inversa sólo nos va a servir para representar gráficamente en el tiempo la señal; posteriormente aplicaremos más filtros y lo haremos en frecuencia, con la señal que sale tras aplicar el filtro H1, sin hacerle la IFFT). Nota: El filtro posee un polo para z = 1 que se corresponde con la frecuencia de 0 Hz. Eso significa que genera un valor infinito en dicha posición el cual habrá que corregir al aplicarlo. Al tratarse de un filtro pasa bajos y estar normalizado, el valor del módulo de dicho filtro en 0 Hz deberá ser "1". También puede optarse por copiar en los 0 Hz el valor que se encuentre en la componente más cercana a esa posición.

$$H1(z) = \frac{(1 - z^{-15})^2}{(1 - z^{-1})^2}$$

6. Figura 4. Aplicar un filtro pasa-altos a la señal del ECG (H3), para lo cual también se ha creado unas adaptaciones de las funciones de transferencia H(z) de los filtros originales del algoritmo de Pan-Tompkins de manera que sirvan con la nueva frecuencia de muestreo. El filtro pasa altos se compone de dos filtros, un pasa bajos (H2) y un "pasa todos", que combinados dan lugar a un pasa altos (H3), que será el que aplicaremos sobre la señal del ECG. Dichos filtros deberán ser introducidos en MATLAB, normalizados, y tras generar el H3, aplicado sobre la señal del ECG en frecuencia que ya teníamos calculada y filtrada con el filtro H1 (no hay que hacer una nueva FFT). Se representará gráficamente esta nueva señal en frecuencia tras aplicar el filtro en el segundo "subplot" (2x1) de la Figura 4, y tras calcular la IFFT dibujaremos en el primer "subplot" (2x1) la señal del ECG en función del tiempo una vez filtrada con H3. Nota: El filtro H2 también posee un polo para z = 1 que se corresponde con la frecuencia de 0 Hz, de forma que procederemos de la misma forma que en el caso anterior. Cuando se corrija este valor, el filtro H3 tendrá valores correctos para todas las frecuencias.

$$H2(z) = \frac{1 - z^{-80}}{1 - z^{-1}}$$

$$H3(z) = z^{-40} - H2(z)$$

7. **Como método alternativo a los pasos 5 y 6**, se podría haber calculado el filtro pasa banda (H4) que resultaría de aplicar los dos anteriores y aplicarlo de una sola vez sobre la señal del ECG, tal y como aparece en el fichero MATLAB proporcionado junto a esta práctica "Prac4_filtros_PT.m", y que no es más que la multiplicación del filtro H1 por el H3:

$$H4(z) = H1(z) \cdot H3(z)$$

8. Figura 5. Calcular la derivada a la señal ya filtrada. Para ello se aplicará una adaptación de la aproximación a la derivada de orden 4 que aparece en el algoritmo original de Pan-Tompkins, la cual se ha modificado para adaptarse a la nueva frecuencia de muestreo, quedando en este caso un filtro de orden 10. Se representará gráficamente esta nueva señal en frecuencia, tras aplicar el filtro de la derivada, en el segundo "subplot" (2x1) de la Figura 5, y tras calcular la IFFT dibujaremos en el primer "subplot" (2x1) la señal del ECG en función del tiempo una vez filtrada con H5.

$$H5(z) = \frac{(5 + 4z^{-1} + 3z^{-2} + 2z^{-3} + z^{-4} - z^{-6} - 2z^{-7} - 3z^{-8} - 4z^{-9} - 5z^{-10})}{110}$$

9. Figura 6 (1). Elevar la señal al cuadrado muestra a muestra, **EN EL TIEMPO**. Para este proceso tomaremos la señal en el tiempo (es decir, tras realizar la última IFFT), que la tenemos calculada del apartado anterior ya que la hemos representado gráficamente tras realizar la derivada, y muestra a muestra la elevaremos al cuadrado consiguiendo que todos los valores queden positivos y que los picos presentes destaquen más todavía sobre el resto de la señal. Se representará gráficamente, en el primer "subplot" (2x1) de la Figura 6, la señal obtenida tras elevar al cuadrado cada muestra en función del tiempo.

$$y[n] = ([x(n)])^2$$

10. Figura 6 (2). Aplicar una ventana de integración a la señal, de tal manera que se sumen los datos a lo largo de toda esa ventana que estará definida para 150 ms, por lo que será necesario calcular previamente cuántas muestras (N) corresponderán a dicha ventana, teniendo en cuenta la frecuencia de muestreo de la señal del ECG (Fs = 500 Hz). En el segundo "subplot" (2x1) de la Figura 6 representaremos gráficamente la señal ya integrada, en función del tiempo. La ecuación de dicha ventana de integración será la siguiente:

$$y[n] = \frac{1}{N}\left[x[n-(N-1)] + x[n-(N-2)] + \cdots + x[(n)]\right] = \frac{1}{N}\sum_{i=0}^{N-1} x[(n-i)]$$

11. Algoritmo de detección del QRS mediante umbrales. Se buscarán los distintos picos máximos que aparezcan en la señal, y se establecerán unos umbrales para detectar si se trata de un pico correspondiente a la señal del QRS (lo llamaremos "señal" a secas) o bien si pertenece a otro tipo de evento, como una onda T o una alteración en la onda (lo llamaremos "ruido"). Estos umbrales serán los siguientes: "SPKI" lo actualizamos cuando encontramos un nuevo pico perteneciente al QRS; "NPKI" lo actualizamos cuando encontramos un nuevo pico que no pertenece al QRS; "THRESHOLD

I1" será el umbral que definiremos para encontrar un complejo QRS (si el pico máximo encontrado lo supera) o de ruido (si ese pico es inferior a dicho umbral). Los valores de "SPKI" y "NPKI" se inicializarán analizando el primer segundo del ECG, de manera que el "SPKI" será 1/3 del pico máximo encontrado en dicho intervalo de tiempo, y el "NKPI" será la 1/2 de la media en el mismo intervalo. A partir de ese instante, los valores se actualizarán para cada pico encontrado de la siguiente forma:

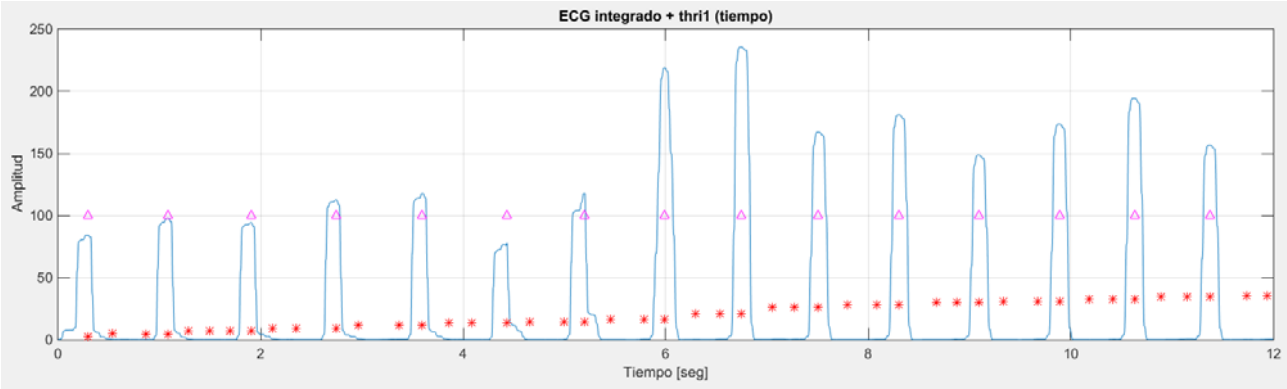
SPKI = 0.125 PEAKI + 0.875 SPKI  si PEAKI es un pico del complejo QRS
NPKI = 0.125 PEAKI + 0.875 NPKI  si PEAKI es un pico de ruido
THRESHOLD I1 = NPKI + 0.25 (SPKI – NPKI)

12. Figura 7. La salida del script serán dos "subplots" (2x1) con las siguientes representaciones: 1º- La gráfica del ECG integrado (arriba), al igual que en la Figura 6, pero donde superpondremos otras 2 gráficas con los valores que va tomando "THRESHOLD I1" cuando éste cambia, y los puntos donde se encuentre un complejo QRS que marcaremos con un símbolo "▲"; 2º- En el "subplot" inferior dibujaremos el ECG sin OFFSET (igual que el de la Figura 1) para poder observar la correspondencia entre los complejos QRS encontrados por el algoritmo y el ECG original. Para dibujar estas gráficas utilizaremos un código similar al siguiente:

```
figure (7); subplot (2,1,1);
plot (t,ECG_integ); grid on; hold on;
thrs_positivos = thrs > 0;
plot(t(thrs_positivos), thrs(thrs_positivos), '*r'); hold on;
```

De esta manera se crearía en la Figura 7 un primer subplot donde se dibuja la señal "ECG_integ" (el ECG tras la ventana de integración), se pone la cuadrícula y se hace un "hold on" para que lo siguiente a dibujar se superponga con la gráfica actual. En la variable "thrs" tenemos almacenados los valores de "THRESHOLD I1" cuando éste ha cambiado durante el desarrollo del algoritmo, y en los instantes donde no ha cambiado se sitúa un 0, de manera que todos los valores distintos de 0 serán los que corresponden a picos máximos encontrados. Así, dibujaremos un "*" en color rojo en todos los puntos diferentes de 2 mediante las dos últimas líneas que se describen (se buscan los valores de "thrs" positivos, y se hace un "plot" sólo de dichos valores). El alumno creará una gráfica con los valores de QRS procediendo de la misma forma que con los "THRESHOLD I1", de forma que se marcarán con un símbolo "▲" solamente los puntos donde se ha encontrado un complejo QRS.

Un ejemplo de aplicación del algoritmo podría quedar como se aprecia en la siguiente figura, donde se ve cómo evolucionan los valores de "THRESHOLD I1" (en rojo) y las marcas (en magenta) de los complejos QRS encontrados:



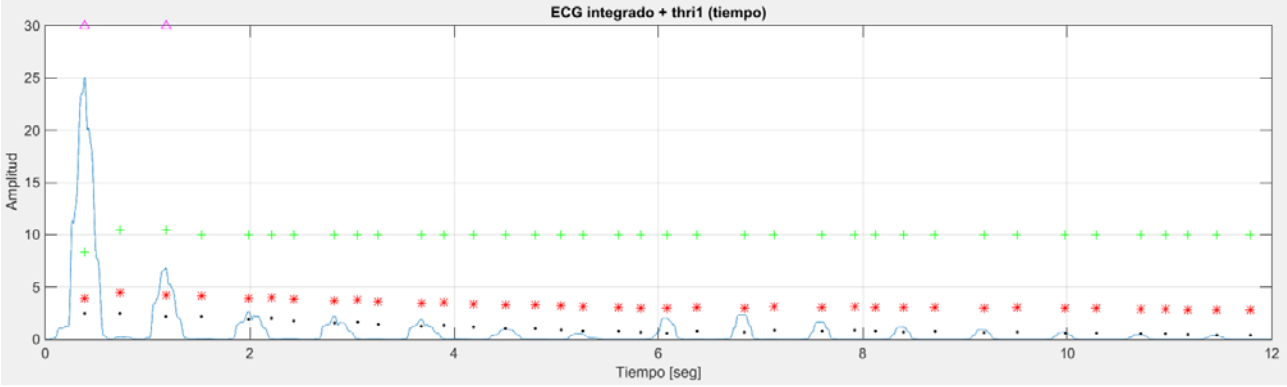

**Notas**:

Hay que tener en cuenta que a la hora de calcular la IFFT en los pasos intermedios para observar cómo queda la señal en el tiempo, el comando "ifft" de MATLAB produce datos con parte real e imaginaria, por lo que para comprobar que la aplicación de los filtros se hace de manera correcta deberéis observar ambas magnitudes (real e imaginaria) por separado en dos subplots de una figura auxiliar, y la componente imaginaria debe ser muy pequeña, del orden de $10^{-10}$ o menos (si se obtienen grandes valores en la parte imaginaria es que algo no se está aplicando de forma correcta). Para la presentación de la figura final que se pide en este enunciado, sólo se mostrará la parte real de la señal.

En el punto 11 del algoritmo se puede usar la función de MATLAB "findpeaks" para buscar los picos máximos de la señal integrada, consultar la ayuda del propio MATLAB para ver cómo utilizarla.

El alumno tendrá que elaborar este primer apartado de MATLAB a partir de los scripts realizados en las anteriores prácticas, el script que se entrega como apéndice a la actual práctica, y los conocimientos de la asignatura. A continuación, contestará a las siguientes preguntas:

**Pregunta 1.1**. A la vista de los resultados del algoritmo Pan-Tompkins, comenta cómo ves su evolución y los problemas que presenta según el tipo de señal de ECG que le introduzcas. Nota: Fíjate sobre todo en el comportamiento con la señal E1000000.MAT y la E1140536.MAT, para ver realmente la ejecución del algoritmo recomiendo dibujar en la Figura 7, además de los cambios en el umbral "THRESHOLD I1", la evolución de los NKPI y los SPKI en todo momento, de manera que si los observas detenidamente lograrás comprender bien cómo funciona el algoritmo dependiendo del tipo de picos (señal o ruido) que se va encontrando. Por ejemplo, la salida del algoritmo podría quedar como se muestra en la imagen siguiente:



Las marcas "+" en verde indican el nivel del "SPKI" en los momentos que cambia, así como los puntos "." en negro indican el "NKPI" y, como vimos antes, los símbolos "*" en rojo marcan el "THRESHOLD I1" y los símbolos "▲" en magenta indican que se ha encontrado un complejo QRS. ¿Qué está ocurriendo en el resto de casos? ¿Cómo evolucionan estos umbrales? ¿Por qué no se detecta más complejos QRS a partir del 2º que se indica? ¿Qué le haría falta al algoritmo para que funcionara de manera más adecuada?

**Pregunta 1.2**. Observa la forma que tienen los filtros pasa bajos y pasa altos en el script de MATLAB que se adjunta a la práctica (H1 y H3), y analiza el efecto que tienen estos filtros que se van aplicando a las señales, centrándote en las frecuencias que nos interesa del ECG. A partir de dichas observaciones, sobre todo comparando las Figuras 2 y 4, comenta si te parecen buenos filtros para el objetivo que nos interesa, que es respetar la señal del ECG entre los 5 Hz y los 15 Hz aproximadamente porque es donde se concentra la mayor parte de la energía del QRS, pero sin llegar a eliminar el resto de información porque también forma parte del mismo, según vimos en teoría:

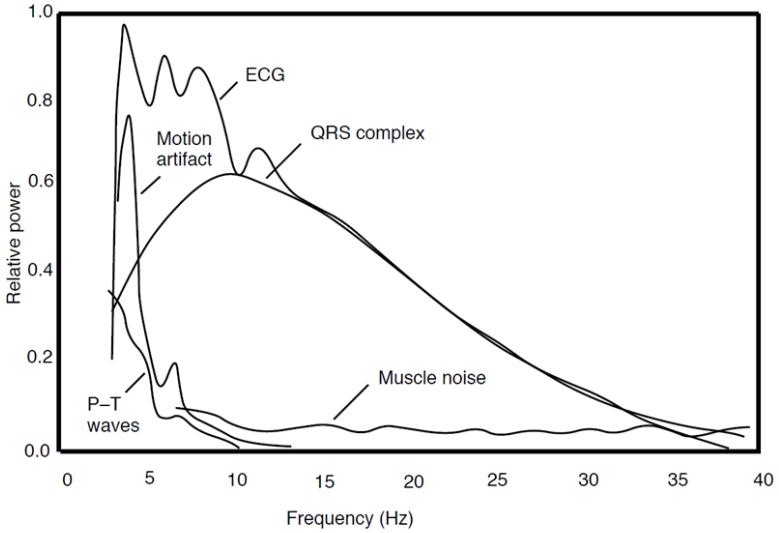



**Figure 12.1** Relative power spectra of QRS complex, P and T waves, muscle noise and motion artifacts based on an average of 150 beats.

**Pregunta 1.3**. Propón una forma de mejorar los filtros usados para el algoritmo de Pan-Tompkins, a través del uso de la herramienta de MATLAB "Filter Design & Analysis", creando tu propio filtro y aplicándolo al algoritmo de detección del QRS en lugar del propuesto en la práctica. Para ello crea una segunda versión de tu script de MATLAB con todo igual, pero con un filtro que deje las frecuencias importantes del QRS y atenúe ligeramente las que son del ECG pero no pertenecen al complejo QRS (o bien que son de otros ruidos presentes en la señal), y que sustituiría a la aplicación de los filtros H1 y H3. Expón las ventajas y desventajas que presenta este filtro frente a los propuestos en el algoritmo original y compara los resultados obtenidos detectando complejos QRS. Para ello muestra alguna captura de la pantalla de una misma señal de ECG procesada con ambos filtros, el de la práctica y el propuesto por ti, indicando en qué mejora tu filtro al de la práctica.

**MEJORAS DEL ALGORITMO**

MEJORA 1 (DIFICULTAD BAJA – OBLIGATORIO DE IMPLEMENTAR). Se debería tener en cuenta que, entre pico y pico máximos de la señal integrada, no pueden transcurrir menos de 200 ms ya que es el ritmo cardíaco más alto posible (correspondería a unos 300 latidos por minuto), por lo que el algoritmo debe descartar los picos más bajos que se encuentren dentro de dicho periodo de tiempo ya que corresponderán a ruido con toda seguridad. Nota: La implementación de esta mejora es necesaria y muy sencilla de hacer, observando cómo funciona el comando "findpeaks" de MATLAB.

MEJORA 2 (DIFICULTAD MEDIA - OPCIONAL). Para reforzar la manera de encontrar un complejo QRS, se puede utilizar un método similar al descrito en el punto 11 pero aplicando otro juego de niveles y umbrales pero sobre la señal del ECG filtrado. En dicho caso, para encontrar un pico correspondiente al QRS debe verificarse que aparece tanto en la señal integrada como en la filtrada, si bien no aparecerán exactamente en el mismo tiempo pero sí en un margen de, como máximo, el tamaño de la ventana de integración (150 ms). La manera de proceder sería buscar un pico correspondiente a un posible QRS en la señal integrada y, una vez localizado, mirar en la señal del ECG filtrado como máximo 150 ms hacia atrás del momento en el que se localiza dicho pico a ver si se encuentra también, aplicando sus umbrales adecuados (cada señal tiene sus propios umbrales); si es así, se verifica que es un QRS, y si no cumple con los requisitos se descarta. Los umbrales para la señal filtrada serían:

SPKF = 0.125 PEAKF + 0.875 SPKF si PEAKF es un pico del QRS en el ECG filtrado
NPKF = 0.125 PEAKF + 0.875 NPKF si PEAKF es un pico de ruido en el ECG filtrado
THRESHOLD F1 = NPKF + 0.25 (SPKF – NPKF)

MEJORA 3 (DIFICULTAD ALTA - OPCIONAL). Técnica de búsqueda hacia atrás. Cuando durante cierto tiempo no se encuentra ningún complejo QRS se aplica una búsqueda hacia atrás aplicando nuevos umbrales menos restrictivos, para lo cual hay que definir unos nuevos intervalos y parámetros.

Un intervalo RR es el tiempo transcurrido entre la detección de 2 complejos QRS consecutivos, es decir, el tiempo que tarda en aparecer una onda R de un latido y la onda R del siguiente. El algoritmo se encargaría de mantener dos intervalos de valores RR junto con la media de los mismos: el primero, RR AVERAGE1, está formado por los 8 últimos intervalos RR encontrados (y su media), mientras que el segundo, RR AVERAGE2, estará formado por los 8 últimos intervalos RR' que se encuentran entre ciertos valores (y su media, también):

RR AVERAGE1 = 0.125 (RRn – 7 + RRn – 6 + … + RRn )
RR AVERAGE2 = 0.125 (RR'n – 7 + RR 'n – 6 + …+ RR 'n )

Los valores de RR'n son los valores de RR que están entre los límites inferior y superior que se describen a continuación:

RR LOW LIMIT = 92% × RR AVERAGE2
RR HIGH LIMIT = 116% × RR AVERAGE2

El algoritmo actuará de la siguiente forma: siempre que no se detecte ningún complejo QRS durante un cierto periodo de tiempo RR MISSED LIMIT, se vuelve hacia atrás y se realiza una nueva búsqueda pero utilizando unos nuevos umbrales (la mitad de los anteriores):

RR MISSED LIMIT = 166 % × RR AVERAGE2
THRESHOLD I2 = 0.5 THRESHOLD I1
THRESHOLD F2 = 0.5 THRESHOLD F1

Si aplicando los nuevos umbrales THRESHOLD I2 y THRESHOLD F2 el algoritmo detecta un complejo QRS en la zona donde antes no se había encontrado, actualizaremos los umbrales de la siguiente forma, adaptándose SPKI y SPKF de manera más rápida al nuevo pico detectado:

SPKI = 0.25 PEAKI + 0.75 SPKI
THRESHOLD I1 = NPKI + 0.25 (SPKI – NPKI)
THRESHOLD I2 = 0.5 THRESHOLD I1
SPKF = 0.25 PEAKF + 0.75 SPKF
THRESHOLD F1 = NPKF + 0.25 (SPKF – NPKF)
THRESHOLD F2 = 0.5 THRESHOLD F1

Por último, el algoritmo sería capaz de establecer si el ritmo de los latidos es normal o no, para lo que deberían coincidir los 8 últimos intervalos de RR con los 8 de RR', esto es, que todos los RR del primer intervalo caigan dentro de los límites inferior y superior definidos anteriormente (RR LOW LIMIT y RR HIGH LIMIT). Si el ritmo de los latidos es normal, los umbrales de detección THRESHOLD I1 y F1 se actualizan con las fórmulas vistas anteriormente. Si por el contrario el ritmo de los latidos no es normal, estos umbrales se reducen a la mitad para no perder ningún latido futuro:

THRESHOLD I1 = 0.5 THRESHOLD I1
THRESHOLD F1 = 0.5 THRESHOLD F1

La salida del algoritmo, por tanto, indicaría tanto el momento en el que se detecta un complejo QRS como si en cada uno de ellos se considera un ritmo normal o no, dibujando una nueva gráfica indicativa de dichos eventos.

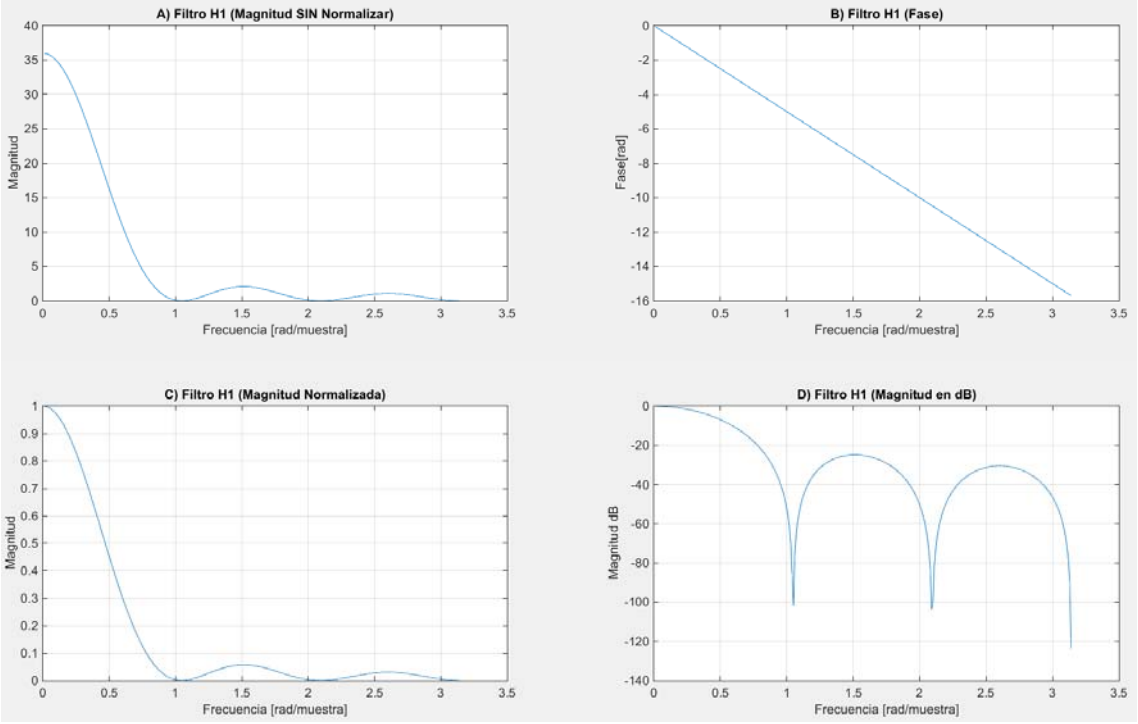
Nota: Las señales de ECG donde se puede aplicar de manera más óptima esta mejora 3 son la "e1000000.mat", la "e113.mat" y la "e222.mat", siendo estas dos últimas de una muy larga duración en el tiempo.

# APÉNDICE: CONVERSIÓN DE FILTROS PARA LA IMPLEMENTACIÓN DEL ALGORITMO PAN-TOMPKINS.

**Prac4_filtros_PT.m**: Se entrega este fichero donde se representan los filtros utilizados en el algoritmo original de Pan-Tompkins, y una adaptación de los mismos para una frecuencia de muestreo mayor del ECG. Para ejecutar el script de modo que aparezcan los filtros originales, habrá que poner el valor de la variable "Pan_Tompkins" a 1, y si queremos los filtros alternativos para una frecuencia de muestreo Fs de 500 Hz situaremos esa misma variable a 0.

La razón por la que hay que convertir los filtros al cambiar de frecuencia de muestreo es porque los filtros están pensados para que tengan los ceros y los polos en determinadas posiciones (en ciertos ángulos en la representación compleja utilizando el diagrama de ceros y polos que conocemos). Al tener el filtro una forma definida en determinados ángulos, si variamos la frecuencia de muestreo de la señal hará que las frecuencias eliminadas sean distintas para cada Fs que tengamos, por lo que el filtro no actuará de la misma forma, o dicho de otra manera, los filtros no son independientes de la frecuencia de muestreo utilizada.

Pongamos un ejemplo. Tenemos el siguiente filtro pasa bajos para el algoritmo de pan-Tompkins original, que posee una Fs de 200 Hz:



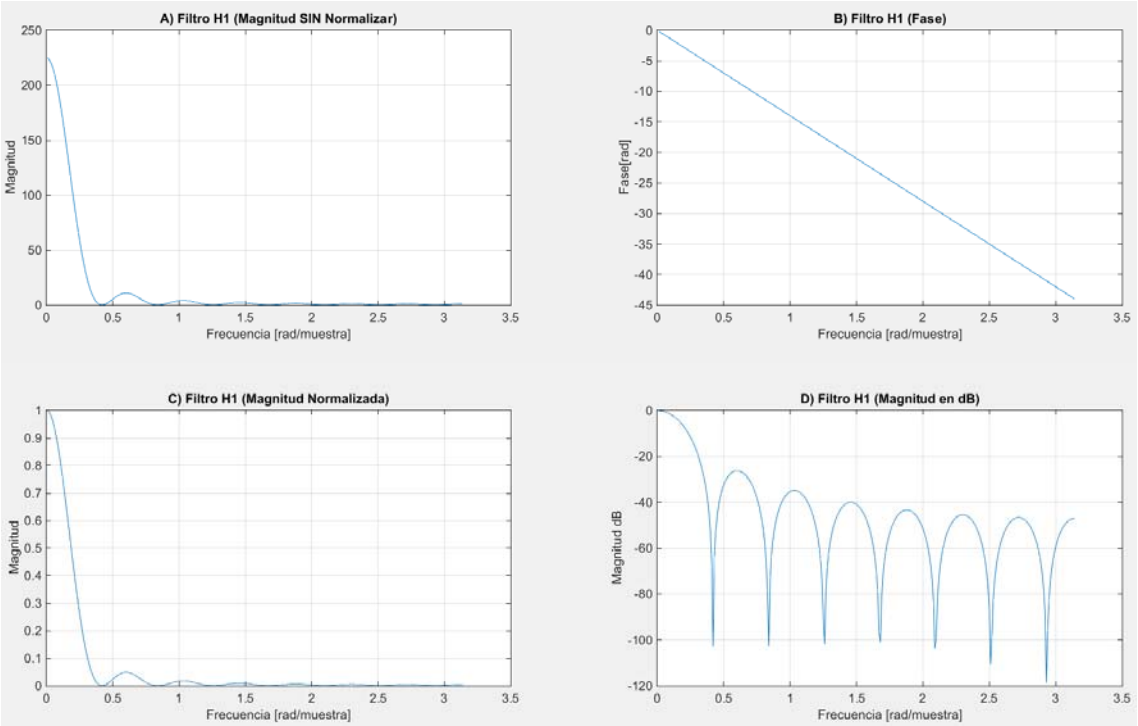

Vemos que posee 3 ceros situados en π/3, 2π/3 y π, mientras que empieza a reducir a la mitad la señal (disminuye a –6 dB o la multiplica por 0.5 si lo miramos en la figura de la escala lineal normalizada) a partir de 0.47 radianes/muestra, lo que equivale a unos 15 Hz utilizando una Fs de 200 Hz como se puede calcular fácilmente:

π rad/muestra → 100 Hz (Fs/2)

0.47 rad/muestra → X

Siguiendo la misma gráfica, observamos una reducción a –12 dB en torno a los 20 Hz (0.64 rad/muestra), y –37 dB al llegar a los 30 Hz (0.94 rad/muestra), lo cual nos indica que a partir de esa frecuencia ya se filtraría mucho todo lo que hubiera en el ECG de entrada (30 dB de reducción ya empieza a ser una pérdida considerable).

Nuestro reto, por tanto, es diseñar un filtro con características similares pero para una frecuencia de muestreo Fs de 500 Hz. Para ello, tomaremos un criterio sencillo que será el de elevar el orden del filtro en 2.5 veces, ya que la nueva frecuencia de muestreo también es 2.5 veces la antigua (500 Hz = 200 Hz * 2.5). Así, situaremos más ceros espaciados en la circunferencia de la representación del diagrama de ceros y polos con una densidad mayor (2.5 veces mayor, obviamente), y obtendremos un filtro con unas características similares al que ya teníamos. En nuestro caso, éste es el resultado para el filtro pasa bajos:




Vemos que presenta 7 ceros (y “medio” que no es posible representar obviamente) lo cual es lo que se esperaba, ya que el filtro original presentaba 3 ceros; empieza a reducir a la mitad la señal (disminuye a –6 dB) a partir de 0.19 radianes/muestra, lo que equivale a unos 15 Hz utilizando una Fs de 500 Hz; presenta una reducción a –12 dB en torno a los 20 Hz (0.25 rad/muestra), y alcanza los –38 dB al llegar a los 30 Hz (0.376 rad/muestra), por tanto tiene unas características muy similares al filtro del algoritmo original.

Para el diseño de los otros filtros adaptados se ha seguido una lógica similar a la anterior, de manera que las gráficas que se recogen tienen las siguientes características:
Figura 1 → Filtro H1 pasa bajos entre 5 Hz y 15 Hz.

Figura 2 → Filtro H2 pasa bajos, que servirá para hacer el filtro pasa altos final.

Figura 3 → Filtro H3 pasa altos, que se obtiene restando de un filtro pasa todo el filtro H2 pasa bajos. Presenta un valor de –6 dB en 4.4 Hz (0.055 rad/muestra) y a partir de ahí deja la señal poco más o menos como estaba.

Figura 4 → Es el filtro H4 pasa banda, que es la composición del filtro pasa bajos H1 y el pasa altos H3 (la multiplicación de ambos en frecuencia). Se aprecia claramente la banda de frecuencia que deja pasar, que sitúa una pérdida de 6 dB sobre los 4 Hz (algo menos de lo deseado, que eran 5 Hz) y en los 14 Hz (también algo menos de lo deseado, que eran 15 Hz). Las atenuaciones en frecuencias más altas se quedarían como en el filtro H1 diseñado, –12 dB en 21 Hz aproximadamente, y –39 dB en 30 Hz.

Figuras 5 y 6 → Se muestra el filtro que implementa la derivada, también adaptado a la nueva frecuencia de muestreo, utilizando el ajuste por mínimos cuadrados explicado en el libro de Tompkins de la bibliografía, páginas 114 a 116. Se ha escogido un filtro de orden 10, cuando el original era de orden 4, siguiendo el esquema explicado en el libro. La representación de la Figura 6 está hecha en forma logarítmica en el eje X (usando el comando "semilogx") y en decibelios en el eje Y, para que encaje con la representación mostrada en el algoritmo de Pan-Tompkins (ver página 255 del libro de Tompkins):