UNIVERSIDAD CENTRAL "MARTA ABREU" DE LAS VILLAS
VERITATE SOLA NOBIS IMPONETUR VIRILIS TOGA. 1948

FACULTAD DE INGENIERÍA ELÉCTRICA

Departamento de Telecomunicaciones y Electrónica

Trabajo de Diploma

"Potenciales Evocados"

Autor: Ahmad Mohamad Mezher

Tutores: Dr. Alberto Taboada Crispi
Ing. Alexander Falcón Ruiz

Santa Clara

2007

"Año 49 de la Revolución"

Hago constar que el presente trabajo de diploma fue realizado en la Universidad Central "Marta Abreu" de Las Villas como parte de la culminación de estudios de la especialidad de Telecomunicaciones y Electrónica autorizando a que el mismo sea utilizado por la Institución, para los fines que estime conveniente, tanto de forma parcial como total y que además no podrá ser presentado en eventos, ni publicados sin autorización de la Universidad.

_____.

Firma del Autor

Los abajo firmantes certificamos que el presente trabajo ha sido realizado según acuerdo de la dirección de nuestro centro y el mismo cumple con los requisitos que debe tener un trabajo de esta envergadura referido a la temática señalada.

_____.

Firma del Tutor

_____.

Firma del Jefe de Dpto.
Donde se defiende el
trabajo.

_____.

Firma del Responsable de
Información Científico-Técnica

La ciencia es como la tierra; sólo se puede poseer un poco de ella.

Jean-Baptiste Poquelin Molière

Agradecimiento:

Con este trabajo culmino una etapa importante de mi vida, quisiera agradecer a todos aquellos que contribuyeron de una forma u otra en mi educación y me ayudaron a alcanzar una meta como esta.

Así doy gracias, primero que todos a Dios, a mi familia y especialmente a mi hermanito Ali (Zizo), a mis tutores Alberto y Alexander que me guiaron siempre y a mis amigos que de una forma u otra formaron parte de mi vida aquí en Cuba.

También doy las gracias a Cuba por haberme dado la oportunidad de estudiar aquí.

Dedicatoria

- *A **Dios**, primero que todo, por ser mi razón de ser.*
- *A **mis padres**, quienes me han enseñado siempre lo correcto para llegar a ser la persona que soy hoy, porque si no fuera por ellos no hubiera llegado nunca. Y también les deseo mucha salud y que dios los bendiga siempre y espero poder siempre ser la persona que ellos sueñan tener.*
- *A **mis maravillosos hermanos Nancy, Mostafa y Ali** , por estar siempre apoyándome durante todos los años que he estado lejos del país para que yo pudiera ser profesional y que les deseo lo mejor del mundo y que los quiero mucho también .*
- *A **mi amigo Zaher (el esposo de Nancy),** por ser como hermano mió durante todos estos años y desearme lo mejor siempre y apoyándome para lo que sea.*
- *A **mis verdaderos amigos,** No quisiera mencionar nombres, pues cometería el grave error de olvidar a alguno que debería mencionar o mencionar a alguno que no debería y eso sería imperdonable.*
- *A **mis tutores,** por su ayuda incondicional.*
- *A todos los profesores que me transmitieron sus conocimientos durante los 5 años.*

RESUMEN

Para evaluar algoritmos de detección de potenciales evocados se requiere tener formas de onda que representen a los mismos. No obstante, no siempre se cuenta con bases de datos apropiadas, ni el modo de adquirirlas usando equipos y procedimientos especiales. En este trabajo se presenta una metodología para obtener la forma de onda de los potenciales evocados a partir de la gráfica de los mismos en un artículo científico. Se hacen pruebas que permiten corroborar la validez del método. Dicha metodología pudiera ser utilizada para la 'adquisición' de otros potenciales bioeléctricos de difícil acceso. Se seleccionaron tres representaciones gráficas de señales de potenciales evocados visuales (PEV) de un estándar y una de señales de potenciales evocados auditivos de tronco cerebral de un artículo y se implementó un programa en MATLAB 7.0 que logró obtener la representación discreta de la señal real (vector con amplitudes correspondientes a los periodos de muestreo).

Luego se escogió una señal de PEV real y se propuso un método basado en un banco de filtros para mejorar la relación señal a ruido. Este se aplicó a la señal después de adicionarle interferencias a 60 Hz y 180 Hz. Se probaron muchos tipos de filtros y se verificó cuáles mejoraban la relación señal a ruido. Para ese tipo de señal, donde el espectro de frecuencia está concentrado en una única zona (la mayor parte de la energía de la señal está en una sola región alrededor de 20 Hz), solo hace falta un filtro paso bajo con frecuencia de corte de 20 Hz. Se demostró que usando un filtro paso bajo de Butterworth de segundo orden con una frecuencia de corte de 20 Hz se mejora bastante, incluso apreciable a simple vista, la relación señal a ruido. Se calculó cada vez que pasamos la señal por un filtro, la relación señal a ruido y se comparó con la relación señal a ruido original, confeccionando una gráfica resumen en EXCEL con los resultados. También se hicieron cálculos para los 3 indicadores más importantes en el estudio de potenciales evocados que son CCR (coeficiente de correlación), SDR (cociente de desviación estándar) y NRR (nivel de ruido residual) y se llegó a conclusiones de gran interés.

ÍNDICE

Introducción

Los potenciales evocados (PE) son fluctuaciones de voltaje en el tiempo, generados por el sistema nervioso en respuesta a un estímulo adecuado. Dependiendo del tipo de estímulo que los provoca pueden clasificarse como potenciales evocados auditivos (PEA), somatosensoriales (PES) y visuales (PEV). Los PE constituyen una herramienta diagnóstica de gran valor para la evaluación funcional del sistema nervioso y las diferentes vías sensoriales. El uso de los potenciales evocados constituye un método que requiere poca o ninguna cooperación por parte del sujeto y que pudiera solucionar algunas de las dificultades de otros métodos. En los últimos años se ha avanzado en forma importante en su introducción para el diagnóstico y evaluación pronóstica de pacientes pediátricos. Una de las aplicaciones clínicas de mayor importancia de los PE es en la pediatría.

Para la realización de este trabajo, se realizaron las siguientes tareas:

1- Realizar revisión bibliográfica sobre potenciales evocados y otros aspectos afines.

2- Implementar pruebas clásicas para la mejoría de la relación señal a ruido.

3- Programar algoritmos en MATLAB7.0 para procesar señales obtenidas en estudios de potenciales evocados.

4- Generar imágenes tiempo-realización de potenciales evocados.

5- Implementar indicadores estadísticos para evaluar el promediado de la señal.

6- Escribir informe de tesis.

Organización de la tesis

Este informe está conformado por tres capítulos. El capítulo 1 está relacionado con la revisión bibliográfica del tema y aborda aspectos tales como los tipos de potenciales evocados, las técnicas tradicionales para la detección de los mismos, así como las dificultades que presentan. Se presentan nuevas alternativas, basadas en la imagen tiempo-realización, para estimar los potenciales evocados.

En el capítulo 2 se explica, paso por paso, la metodología seguida, incluyendo como se implementó el programa graph2vector, con el que obtuvimos las señales discretas de potenciales evocados a partir de sus representaciones gráficas seleccionadas de artículos

científicos. Se muestran los resultados obtenidos con las señales reales para cuatro gráficos de amplitud vs. tiempo de dos tipos de potenciales evocados (PEA y PEV). Luego se escogió una de las señales reales obtenidas y se le adicionó una interferencia a 60Hz y otra a 180Hz y se propuso un método basado en banco de filtros para la mejoría de la relación señal a ruido. Se demostró, después de pasar la señal por diferentes tipos de filtros de diferentes órdenes, que para ese tipo de potencial evocado se puede lograr mejora de 12dB en la relación señal a ruido (de 24dB a 36dB) usando un filtro Butterworth de segundo orden.

En el capítulo 3, primeramente se hacen simulaciones de señales de potenciales evocados visuales para obtener imágenes tiempo-realización y evaluar algoritmos de detección-estimación. Un trabajo futuro sería demostrar con estas bases de datos de imágenes tiempo-realización que, aplicando procesamiento de imágenes, con pocas repeticiones se llega a un resultado satisfactorio. En una de las simulaciones se muestra el espectro de frecuencia de la señal y se comprueba que está por debajo de los 40Hz, lo que se corresponde con lo que decía el artículo de donde sacamos las imágenes.

Cabe señalar que en las simulaciones realizadas, se tomaron en cuenta algunos parámetros como el *trigger* (corrimiento relativo de los potenciales), *shimmer* (modulación de amplitud), y *width modulation* (variación del ancho). La modulación de amplitud, por ejemplo, es de gran importancia y se debe en la práctica a la concentración del paciente, o sea, mientras más concentrado el paciente está, mayor es la amplitud, por lo que mientras más dure la sesión, más se cansa el paciente y más se disminuye la amplitud. También se le adicionó a la señal un ruido y una interferencia a 60Hz y otra a 180Hz, obteniendo nuevamente imágenes tiempo-realización ruidosas.

También en el capítulo 3 se evaluaron cuatro métodos de promediado (promediado simple, promediado recortado, mediana y promediado recortado modificado), estimando valores de tres indicadores estadísticos: CCR (coeficiente de correlación), SDR (cociente de desviación estándar), y NRR (nivel de ruido residual).

Finalmente se expresan las conclusiones a que se arriban en este trabajo y se dan recomendaciones para trabajos futuros.

CAPÍTULO I: Estudios de Potenciales Evocados

Se entiende por potencial evocado (PE) la respuesta neuroeléctrica del sistema nervioso ante un estímulo. La forma de realizarlos depende de la vía sensorial que se quiere explorar, por lo que hay distintos tipos de potenciales evocados. En casi todos se colocarán unos electrodos de registro en el cuero cabelludo, estos pueden ser superficiales, pegados con una pasta conductora, o agujas muy finas, que se colocan bajo de la piel. Dependiendo del tipo de potencial, también se pueden poner electrodos en otros puntos del cuerpo [13].

Para realizar esta prueba es necesario que el paciente acuda con el pelo lavado, sin llevar fijador, laca o cualquier otro producto cosmético. No debe venir en ayunas, se puede desayunar o comer normalmente. En los estudios visuales, si el paciente usa espejuelos, debe también usarlos durante la exploración [32].

La determinación de estos PE es de gran interés clínico y diagnóstico ya que permite establecer, por comparación con las respuestas consideradas normales, diversas patologías o disfunciones de las vías nerviosas. Para su obtención, se miden las tensiones eléctricas entre electrodos ubicados en posiciones de la cabeza seleccionadas especialmente para cada tipo de estudio [13] [18].

Estas tensiones, que son el resultado de la actividad neurológica, son enormemente atenuadas por los diversos tejidos (óseo, muscular, epitelial, etc.) que separan el punto donde se originan los potenciales de aquel donde se miden, reduciéndose así a unos pocos micro voltios. El problema clásico de las señales de tan bajo nivel es que están muy expuestas a la interferencia de ruido eléctrico de diversos orígenes, que dificultan su identificación. A la respuesta neuroeléctrica que se desea medir, se superponen los potenciales generados por la actividad muscular y otros potenciales generados por campos externos (efecto “antena” o acoplamiento capacitivo, por ejemplo la captación del campo eléctrico de los tubos fluorescentes o de las líneas de alimentación), o por pequeñas cargas de electricidad estática. Estos ruidos suelen ser comparables o incluso mayores que la propia señal a medir. Se plantea así el problema de rescatar una señal contaminada por ruido [13].

La solución clásica consiste básicamente en repetir el estímulo varias veces y obtener el promedio de las respuestas evocadas. Esto parte de la premisa de que la respuesta neuroeléctrica para un estímulo no varía cuando el estímulo se repite, y en cambio los

valores de ruido que se agregan en cada repetición fluctúan aleatoriamente con valores positivos y negativos que tienden a compensarse, es decir, que su promedio tiende a cero [13] [20] [25].

1.1 Tipos de PE

Los potenciales evocados pueden ser visuales, auditivos o somatosensoriales, atendiendo al estímulo aplicado para obtenerlos.

Visuales: sirven para explorar el funcionamiento del nervio óptico y las vías visuales cerebrales. El paciente mira una pantalla de ordenador con unos cuadros cambiantes o una luz en forma de flash [29] [34].

Auditivos: estudian la vía auditiva hasta el cerebro. Se colocan al paciente unos auriculares mediante los cuales escuchará un sonido a distintas intensidades. Las respuestas se registran con unos electrodos pegados al cuero cabelludo y lóbulos de las orejas [29].

Somatosensoriales: estudian las vías de las sensaciones cutáneas desde la mano o el pie, pasando por los nervios, la médula espinal hasta llegar al cerebro. Se estimula la piel con corriente eléctrica, que el paciente siente como una especie de latidos o golpes rápidos en la muñeca o el tobillo. También puede estimularse en otros niveles, en función de los nervios a explorar [16] [28].

En la actualidad estas pruebas no pueden sustituirse por otras que proporcionen una información similar sobre la función de las vías y los centros nerviosos que se estudian [29].

Los PE se han convertido junto a las técnicas de imaginología en una herramienta de interés para la práctica médica al detectar alteraciones en el Sistema Nervioso, contribuyendo en algunos casos a la localización topográfica de la lesión. Tienen numerosas ventajas [5]:

- Permiten evidenciar cambios que aún no se expresan clínicamente.
- Demandan pocos recursos y se realizan en un breve tiempo.
- No son técnicas invasivas, ni cruentas.

La latencia expresa la velocidad de procesamiento de información, y su amplitud indica la concentración de la atención de los sujetos en estudio. La latencia puede ser medida de forma absoluta o entre los picos [5] [20]:

- **Latencia absoluta**: es medida generalmente al pico, aunque también se puede medir al inicio del componente. Refleja la conducción desde el instante de tiempo "cero" (el momento en el que se le presenta estímulo al sujeto) hasta el punto de máxima amplitud del componente y es expresada en milisegundos.
- **Latencia Inter-picos**: resulta de la diferencia de latencia entre dos picos, medidos en aquellas respuestas que se caracterizan por la presencia de más de un pico.

1.2 Características Generales de los PE

Con el fin de diseñar un sistema capaz de registrar PE para su posterior procesamiento, se deben tomar en cuenta características tales como amplitud, contenido de frecuencias, tiempo de presentación de estímulos, ocurrencia simultánea de otras señales EEG, ruido de instrumentación y variaciones de la señal con respecto al tiempo [32].

Las amplitudes de los PE van de las décimas de micro-voltio a decenas de micro-voltios (1-40) y, por convención, las amplitudes positivas se consideran negativas y viceversa. De acuerdo a esta convención y considerando la latencia, las amplitudes características se designan como N1 (N100), P1 (P100), P3 (P300), etc. Estas amplitudes juegan un papel muy importante en el análisis de los PE [32].

El hecho de que el PE se encuentre sumergido en una infinidad de señales, las cuales se considerarán como ruido, conduce a situaciones en las que se tiene una relación del voltaje del PE con respecto al voltaje de EEG, por ejemplo, de 1:100, lo que corresponde a una relación de -40 dB. Esta relación es tan pequeña que hace que la detección y estimación del PE sea muy difícil [32].

El PE representa la respuesta del cerebro al estímulo en el orden de los milisegundos y para su estudio, se divide en tres intervalos de tiempo, los cuales indican diferentes tipos de información. En el primer intervalo se encuentran los eventos tempranos (1,5 - 10 ms) los cuales reflejan la transmisión de información aferente a través de varios niveles en la trayectoria sensorial. EI segundo intervalo comprende los eventos medios (50 - 200 ms) los cuales reflejan la llegada de información aferente a la corteza específica ya sea sensorial o no sensorial y permiten observar aspectos funcionales del cerebro como la agudeza sensorial, atención, etc. Por último, se tienen los eventos tardíos (200 - 500 ms), entre los cuales se encuentra el P300, al cual se le ha asociado una gran variedad de procesos

cognoscitivos complejos. Los engloba en 3 dimensiones que son las siguientes: probabilidad subjetiva, significado del estímulo y transmisión de información [32].

Se puede hacer una clasificación de los potenciales en dos clases. Aquellos que son debidos a un estímulo físico externo llamados exógenos y aquellos que se relacionan con procesos psicológicos y que reciben el nombre de endógenos o emitidos. Los potenciales exógenos son los PE visuales, auditivos y somatosensoriales que se producen directamente por la aplicación del estímulo, mientras que los potenciales endógenos se identifican como procesos internos entre los cuales se encuentran los potenciales de intención o preparación que proceden a un movimiento voluntario. También está el CNV (Contingent Negative Variation) u onda de expectancia, que es aquélla que se presenta entre un estímulo condicional y uno imperativo o incondicional, con el cual el sujeto está de alguna manera ligado. También se presentan potenciales imaginarios o respuestas a un estímulo que se esperaba y que fue omitido [32].

1.3 Ruidos que Afectan los PE

Hay 2 tipos de ruido en potenciales evocados. Por un lado se encuentran las señales EEG (potenciales electroencefalográficos) y ECG (potenciales electrocardiográficos), cuya presencia es permanente e inevitable, y por otro lado los potenciales de origen interno y externo que pueden reducirse o minimizarse tomando ciertas precauciones. Estos últimos reciben el nombre de artefactos. Algunos de estos tienen también carácter permanente, como el ruido de 60Hz de la línea de alimentación, que aparece ya sea por un mal aislamiento, como por radiación de transformadores o tubos fluorescentes. Otros, son fortuitos, como los potenciales asociados a la actividad muscular. Así un pestañeo, o la contracción de otros músculos, especialmente aquellos que se encuentran próximos a los electrodos, pueden producir picos de ruido bastante importantes. Los artefactos de nivel similar al resto de la señal no merecen un tratamiento particularizado. Aquellos artefactos de magnitud desmedidamente grande, en cambio, pueden alterar significativamente el promedio. En estadística esto se denomina un valor fuera de control y lo aconsejable es descartarlo. En los promediadores se suele proveer una función que permite descartar las respuestas en las que aparecen niveles mayores que determinado límite. Este nivel puede ajustarse en función de lo que se considere una respuesta razonable para cada tipo de estudio [13] [20].

1.4 Pruebas Clásicas de PE

1.4.1 Promediado

Para comprender mejor la forma en que se realiza el promediado, veamos un ejemplo. Supongamos que nos interesa obtener la respuesta durante los 10 ms siguientes al comienzo del estímulo, y que nos interesa una precisión en el tiempo de 0,1 ms. Entonces, para cada repetición de estímulo medimos y anotamos el valor del potencial evocado cada 0,1 ms, vale decir, en los instantes 0 ms, 0,1 ms, 0,2 ms, …., 9,9 ms y 10 ms desde el comienzo del respectivo estímulo. Esto significa que para cada repetición del estímulo tomamos 101 muestras de dicho potencial. Así, si decidimos efectuar 100 repeticiones (más adelante se indica como se selecciona la cantidad apropiada de repeticiones), tendremos un total de 10100 muestras. Entonces se calcula el promedio de los 100 valores correspondientes al instante 0,1 ms, luego se calcula el promedio de los 100 que corresponden a 0,2 ms, etc. el resultado de esos promedios es una buena estimación del verdadero potencial evocado en los respectivos instantes de tiempo [13] [18].

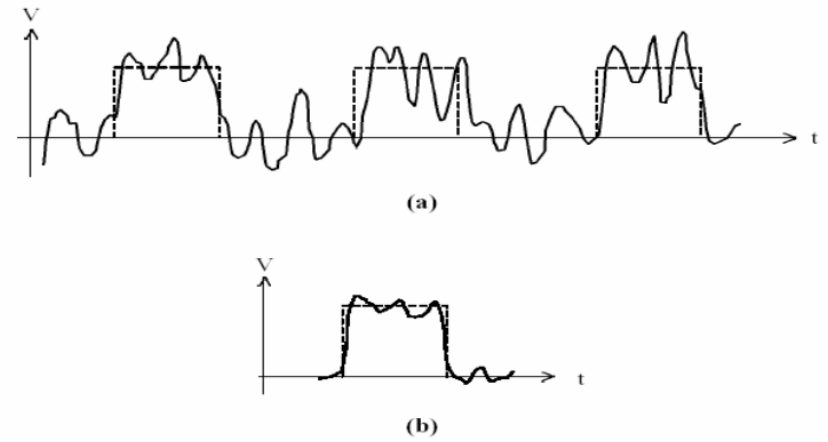

Figura 1. (a) Tres repeticiones sucesivas de un mismo estímulo, con la misma respuesta (un pulso rectangular, en línea de trazos) contaminada por ruidos diferentes. (b)El resultado de promediar las tres respuestas contaminadas: una onda más fiel a la respuesta pura.

Selección de la Cantidad de Estímulos

La cantidad de veces que se repite el estímulo depende del nivel de ruido existente. Intuitivamente, si no hubiera ruido, un solo estímulo sería suficiente. A medida que aumenta la proporción de ruido (cualquiera sea su origen) con respecto a la señal (el potencial Evocado que se desea medir), será necesario incrementar la cantidad de estímulos, ya que de esa forma el error originado en el ruido se va “diluyendo” [13] [18].

A partir de razonamientos estadísticos se concluye que la cantidad de estímulos requeridos Para mejorar la calidad de la señal en una determinada proporción aumenta con el cuadrado del nivel de ruido. Para expresarlo de una manera más precisa, definamos primero la relación señal a ruido (S/R) como el cociente entre el nivel de la señal y el nivel del ruido, ambos en micro voltios. Así, si el nivel de la señal es de 10 μV y el del ruido de 5 μV, entonces la relación señal a ruido es

$$S/R = \frac{10\ \mu V}{5\ \mu V} = 2\ .$$

Ahora supongamos que la relación señal a ruido antes del promediado es S/R_1, y la relación señal a ruido que se desea obtener es S/R_2. Entonces el número n de estímulos requerido es:

$$n = \left(\frac{S/R_2}{S/R_1}\right)^2 .$$

El valor de S/R_2 normalmente está dictado por la mínima variación de la señal que tiene importancia para una interpretación confiable de los resultados. Así, por ejemplo, si el nivel de la señal es de 10 μV, y existe una onda de 2 μV que tiene importancia para el estudio que se está realizando, el ruido deberá reducirse a menos de 1 μV para que la aparición o no de esa onda sea atribuible a la respuesta neuroeléctrica y no al ruido. Tomando entonces una cota de 0,5 μV, por ejemplo, resultará

$$S/R_2 = \frac{10\ \mu V}{0{,}5\ \mu V} = 20\ ,$$

de donde, tomando como base el ejemplo anterior, es decir $S/R_1 = 2$, resultará

$$n = \left(\frac{20}{2}\right)^2 = 100\ .$$

Limitaciones de las Pruebas Clásicas

Este sencillo ejemplo (lo de la selección de la cantidad de estímulos) muestra cuán rápidamente crece el número de estímulos requerido para depurar la señal a un nivel suficiente para posibilitar una interpretación confiable de lo observado. Sería deseable, por lo tanto, aplicar una cantidad de estímulos muy grande, pero existen limitaciones de orden práctico para ello. En primer lugar, cada estímulo requiere un tiempo que según el estudio a realizar puede variar entre 50 y 1000 ms (o más), lo cual haría la sesión demasiado extensa y agotadora para el paciente. En segundo lugar, la incomodidad que esto generaría en el paciente provocaría diversos movimientos musculares que agregarían aún más ruido (artefactos). En la práctica el número de estímulos queda limitado de manera que la sesión no dure más de algunos minutos. Se llega así a un compromiso entre precisión y practicidad [13].

1.4.2 Pre-procesamiento de la Señal

Dado que el nivel de ruido incide cuadráticamente en la cantidad de respuestas requeridas, es preciso reducirlo lo más posible antes de realizar el promediado. A menudo esta reducción puede lograrse mediante un adecuado manejo y acondicionamiento de la señal [13].

Una primera regla es la utilización de amplificadores diferenciales de entrada de alta calidad, o sea, que no agreguen ruido propio apreciable, y que tengan un alto rechazo a las señales de “modo común”, que son los potenciales espurios (no deseados) que se suman a ambos electrodos simultáneamente. De este tipo suelen ser, predominantemente, los ruidos debidos a la línea de alimentación, los tubos fluorescentes y otros dispositivos eléctricos de uso común [13] [18] [21].

Una segunda estrategia consiste en aprovechar que muchos ruidos pueden separarse de la señal útil por aparecer en una banda de frecuencias diferente de la de la señal. Recordemos el hecho de que toda señal que varía en el tiempo puede descomponerse en componentes de diversas frecuencias, denominada espectro de frecuencias o simplemente espectro de la señal, y puede realizarse por medios de filtros. Los filtros son dispositivos que permiten el paso de ciertas frecuencias y bloquean otras. En PE se usan 3 tipos de filtros: los filtros pasa altos, los filtros pasa bajos, y los filtros *notch* (supresores de banda estrecha) [18] [13].

Los filtros pasa altos permiten pasar todas las componentes de frecuencias superiores a una frecuencia F(inf), denominada frecuencia inferior de corte, bloqueando el paso a las componentes de frecuencias menores que F(inf). Dado que las componentes del potencial evocado siempre son mayores que la frecuencia con que se repiten los estímulos, pueden eliminarse todas las componentes de baja frecuencia, reduciendo así ruidos de baja frecuencia provenientes del EEG, del ECG y de la actividad muscular [13] [18].

Los filtros pasa bajos, por el contrario, dejan pasar las bajas frecuencias, hasta una frecuencia F(sup), frecuencia superior de corte, bloqueando las frecuencias que exceden dicho límite. En PE cumplen 2 funciones, la primera es eliminar el ruido de alta frecuencia que se encuentra fuera de la banda de interés, y la segunda satisfacer un requisito de todo sistema muestreado, que es que la máxima frecuencia que ingresa al sistema debe ser menor que la mitad de la frecuencia de muestreo. En el ejemplo en que se tomaban muestras cada 0,1 ms, es decir que se muestreaba a razón de 10000 muestras por segundo, la máxima frecuencia admisible era por lo tanto 5000 Hz, o 5 kHz [13] [18].

Finalmente los filtros *notch* eliminan una frecuencia específica, dejando el resto del espectro prácticamente inalterado. Se utilizan para bloquear por ejemplo la frecuencia de 60Hz de la línea de alimentación (en algunos países, 50Hz) cuya presencia suele ser inevitable cuando están en juego de niveles de señal tan bajos como los que se miden en los PE. El filtro *notch* es el más empleado, aunque no se recomienda en los PE somatosensoriales donde la señal de interés tiene componentes de esta frecuencia, por lo que sólo debe ser usado cuando en algún registro esté presente marcada interferencia de la línea que le imposibilite el trabajo [13] [18].

Los 3 filtros pueden usarse uno a continuación del otro, es decir en cascada. El resultado de este procesamiento previo de la señal es una reducción importante del ruido con la consecuente reducción del número de estímulos requeridos [13] [18].

1.5 Aspectos Generales en los Estudios de PE

Los equipos de PE deben disponer de un conjunto de estimuladores que permita una amplia exploración de los sistemas auditivo, visual, somestésico y motor. El estímulo es seleccionado según la vía a estudiar y la respuesta quedará determinada si son debidamente

manipuladas ciertas características físicas del estímulo tales como: frecuencia de estimulación. Debe ser breve, intenso, controlado y reproducible [5].

Como vimos hasta aquí, se obtiene un buen registro cuando se mejora la relación señal ruido. Para ello, la actividad es sometida a muchos pasos, pero finalmente los dos primeros elementos son decisivos: la cooperación del paciente y la colocación de los electrodos, pues ambos influyen sobre el tiempo que nos tome el registro y la calidad con que se obtengan los PE y por ende la confiabilidad a la hora de la evaluación y emisión de un criterio por parte del médico [5].

Durante el registro, algunos aspectos le serán de utilidad para considerar que la respuesta que se está obteniendo se corresponde con el PE del paciente [5]:

1. Presencia de picos u ondas que caracterizan la respuesta según la modalidad. La ausencia de las mismas puede ser por problemas técnicos o lesión de la vía.
2. Replicabilidad: es la similitud o igualdad entre dos o más registros consecutivos o bien mediante el criterio de hemi-promedios. Cuando esto no sucede puede obedecer a problemas técnicos o a afección de la vía explorada.
3. Indicadores estadísticos de calidad de señal: como su nombre lo indica son estadígrafos aplicados para la evaluación cuantitativa de la calidad de respuesta evocada o para detectar la presencia del PE.

Dentro de los indicadores estadísticos tenemos los siguientes [5]:

- Coeficiente de Correlación (CCR): se calcula como el coeficiente de correlación entre los dos hemi-promedios de un PE. Para el cálculo del CCR se utiliza toda la ventana de tiempo del PE promedio. Este indicador cuantitativo evalúa la similitud en forma de los dos hemi-promedios, es decir, la replicabilidad del PE. Mientras más cercanos a uno sean los valores del CCR más replicable es el potencial.
- En registros obtenidos a altas intensidades, un artefacto de estímulo de gran amplitud puede provocar valores espurios anormalmente altos del CCR aún en el caso de que la señal no sea replicable o que no haya señal.
- Cociente de desviación estándar (SDR): se utiliza como un indicador cuantitativo de la relación señal – ruido de un registro. Se calcula como el cociente entre la desviación

estándar del PE promedio (señal) y la desviación estándar del ruido. Por tanto, mientras mayor sean los valores de este indicador mejores niveles de relaciones señal estándar en el registro. Una frontera aceptable para el SDR serían valores iguales o mayores a dos.

- Nivel de ruido residual (NRR): es un indicador cuantitativo que se utiliza para estimar la energía en micro-voltios del ruido residual en un PE promedio. Para calcular este indicador se suman con signos alternos los registros individuales sucesivos de un PE. De esta manera se elimina la parte constante del PE (señal). Con el registro resultante (ruido) se calcula la desviación estándar como un estimador de la energía media de dicho ruido residual. El NRR puede utilizarse como un criterio para evaluar la calidad del registro. En un registro de calidad aceptable los valores del NRR deben ser menores que la amplitud de la señal. En el caso de los PE auditivos de corta latencia que son respuestas de poca amplitud (aproximadamente de 0,5 a 1 μV) una frontera aceptable para el NRR, serían valores menores o iguales que 0,5 μV.

Para incrementar la resolución de las ondas durante el registro, se debe [5]:

- Aumentar el número de promediados.
- Lograr que el paciente se relaje mejor.
- Mejorar la ubicación de los electrodos de registro a partir de los referidos sitios de generación de las ondas.
- Mejorar la colocación del estimulador.
- Revisar la salida del estímulo, incrementado discretamente la intensidad del mismo por ejemplo en el estímulo eléctrico.

Las mediciones de las variables se evalúan por [5]:

- Comparación de los valores de cada parámetro con los de la norma.
- Comparación de la morfología, así como de los parámetros de latencia y amplitud entre ambos lados.

1.6 Potencial Evocado Auditivo

El proceso indicado anteriormente comenzó a hacerse técnicamente practicable con el advenimiento de las modernas computadoras de gran velocidad y capacidad de memoria, y es a partir de entonces que la técnica de los potenciales evocados auditivos ha ido ganando terreno en las clínicas audiológicas y neurológicas.

Debe observarse que la computadora no cumple únicamente la función de realizar el promedio, sino que además es quien da la señal de sincronismo al generador de sonido para cada estímulo. Otras tareas auxiliares pero no por ello menos importantes consisten en permitir la graficación e impresión de las curvas de potenciales evocados, llevar archivos con historias clínicas, etc. En la Figura 2 se muestra el diagrama de bloques de un instrumento para obtención de potenciales evocados [13].

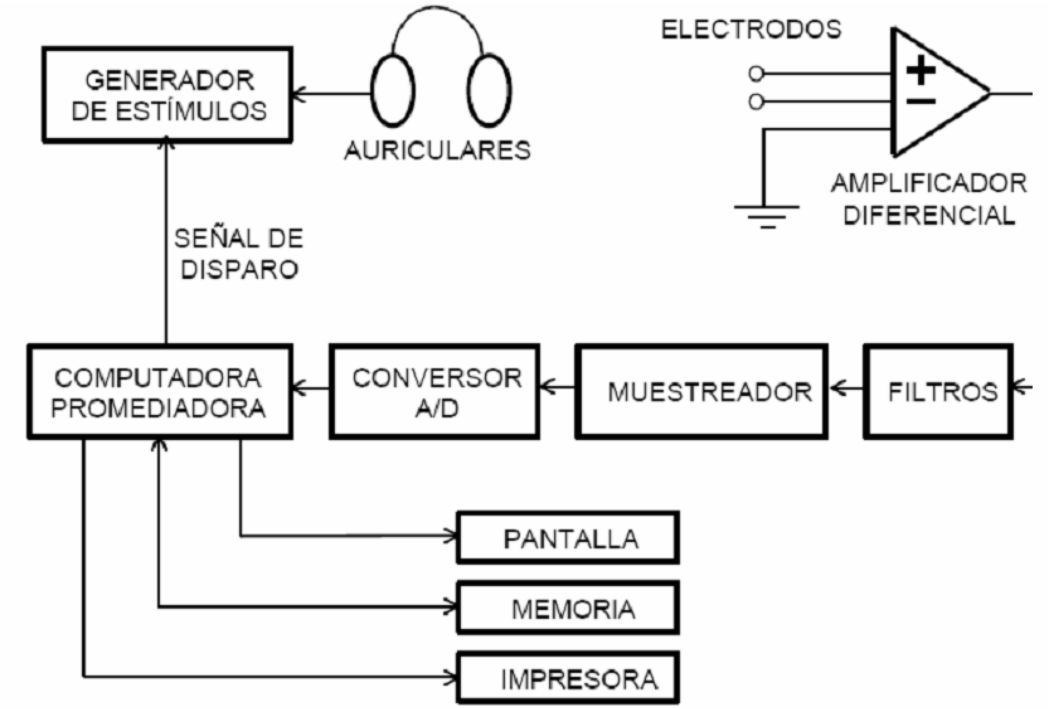

Figura 2. Diagrama de bloques de un sistema de medición de potenciales evocados auditivos.

1.6.1 Selección del Tipo de Estímulo

La selección del tipo de estímulo a utilizar es una de las cuestiones más complejas de todo el proceso de medición del potencial evocado. El estímulo ideal debería permitir, entre otras cosas, determinar objetivamente el umbral de audición a las diversas frecuencias de la

audiometría subjetiva tradicional. Esto es muy difícil de lograr en la práctica ya que los estímulos para PE deben satisfacer dos requisitos que en la práctica se contraponen. En primer lugar deben ser de muy corta duración, ya que debido entre otras cosas al potencial microfónico coclear (ver próxima sección), la presencia del estímulo ocasiona un artefacto que interfiere con el potencial a investigar. Además, un estímulo prolongado tiende a producir un fenómeno de adaptación, que altera considerablemente el perfil del potencial evocado. En segundo lugar, estos estímulos deberían poseer una gran especificidad tonal, lo cual desde el punto de vista espectral implica que la energía debería estar concentrada en una región muy angosta del espectro [13] [20].

Tal como se indicó, estos requisitos se contraponen, ya que las señales de muy corta duración tienden a tener un espectro muy extendido, y las de espectro angosto requieren una duración considerable [13].

A lo anterior se agrega el hecho de que por más que se dispusiera de un estímulo muy corto y de gran especificidad tonal, la cóclea reacciona tonotópicamente sólo en régimen permanente o estacionario. El régimen que imponen los estímulos muy cortos es necesariamente transitorio, siendo difícil establecer una correlación directa entre el PE y el umbral de audición para una frecuencia determinada. Esto es válido muy especialmente para las bajas frecuencias [13] [20].

Para los estudios de PE se utilizan normalmente tres tipos de estímulos: el *click*, el *tone burst*, y el *logon*. El *click* (Figura 3) es una señal que se separa durante un pequeño intervalo del nivel de reposo y luego retorna al mismo. Mientras dura el pulso, el nivel es constante. Cuanto más corto sea el pulso más extenso será el espectro, es decir que la energía sonora se reparte en un rango más amplio de frecuencias [13] [20].

Así, un *click* muy corto permite estimular toda la cóclea. En la Figura 3 se muestra la forma de onda de un click con su espectro de frecuencia.

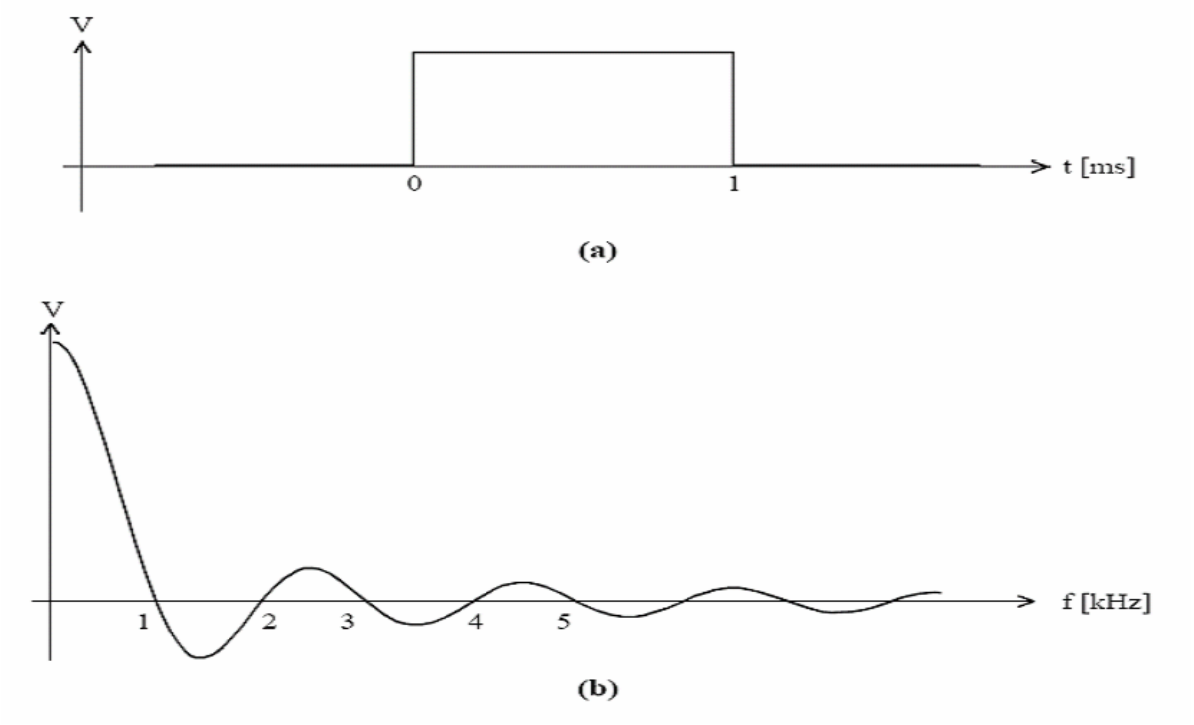

Figura 3. (a) Click de 1 ms de duración. (b) Espectro de frecuencias. Obsérvese que aparece una gran cantidad de energía en las bajas frecuencias.

El *tone burst* (Figura 4) consiste en un tono puro (sinusoidal) limitado a un pequeño número de ciclos. Podría definirse como una sinusoide modulada en amplitud por un *click*. Tiene más especificidad tonal que el *click*, aunque contrariamente a lo que podría creerse, no contiene sólo una línea espectral de la frecuencia del tono puro, sino que se extiende tanto más cuantos menos ciclos dure el *tone burst*. Así, por ejemplo, un *tone burst* de 2 kHz que contenga sólo un semiciclo, se parecerá espectralmente más a un *click* que a un tono puro. Para evitar saltos bruscos derivados de una conmutación que no coincida con un pasaje de la sinusoide por 0, se suele utilizar la técnica denominada *windowing* ("enventanado"), por la cual se reemplaza la modulación con un *click* por la modulación con una onda en forma de trapecio o similar, que asegura una transición más gradual [13].

El *logon* (Figura 5) es un tono puro modulado por una campana de Gauss. Es una forma especial de *windowing*. Su espectro es también una campana de Gauss, que tiene la particularidad de que se reduce muy rápidamente fuera de su zona central, por lo cual la energía se concentra en dicha zona. Por esta razón se logra una buena especificidad tonal aún con un estímulo corto. Se utiliza por ser un buen compromiso entre corta duración y especificidad tonal [13].

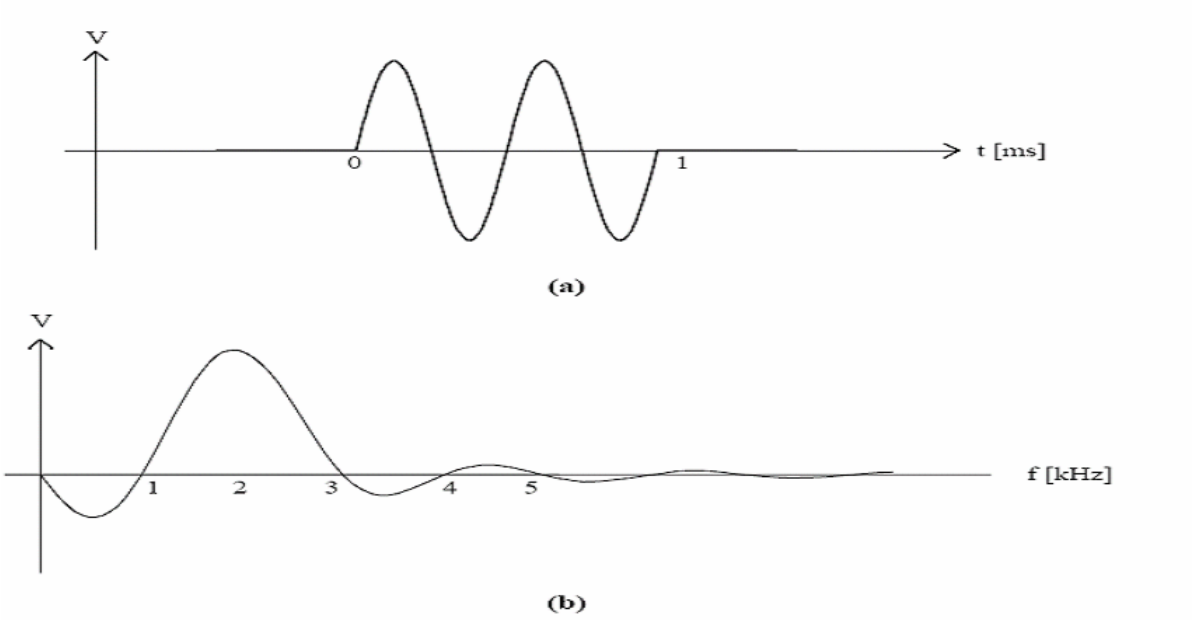

Figura 4. (a) Tone burst de 1 ms de duración, formado por 2 ciclos de una onda sinusoidal de 2kHz. (b) Espectro de frecuencias. Obsérvese que si bien la mayor parte de la energía se concentra cerca de los 2kHz, hay también energía en otras frecuencias, tanto menores como mayores.

1.6.2 Potencial Microfónico Coclear y Potencial de Acción

El potencial microfónico coclear es un potencial que se genera en la cóclea por la vibración de las células pilosas de Corti como respuesta a un estímulo sonoro, en forma similar al potencial generado por un micrófono (de allí su nombre). Es aproximadamente proporcional a la presión sonora recibida en el tímpano, es decir que su forma de onda reproduce la de las ondas de presión sonora. En particular es sensible a la polaridad de dicha presión. Así, si una presión hacia adentro del tímpano (compresiva) produce un microfónico coclear con una polaridad, una presión hacia afuera (descompresiva) producirá un microfónico con polaridad opuesta. El potencial de acción, en cambio, es la respuesta de una neurona, y como tal se produce al superarse un umbral. Por consiguiente, su forma no depende esencialmente de la forma de onda de la excitación, y su polaridad es constante [13] [20].

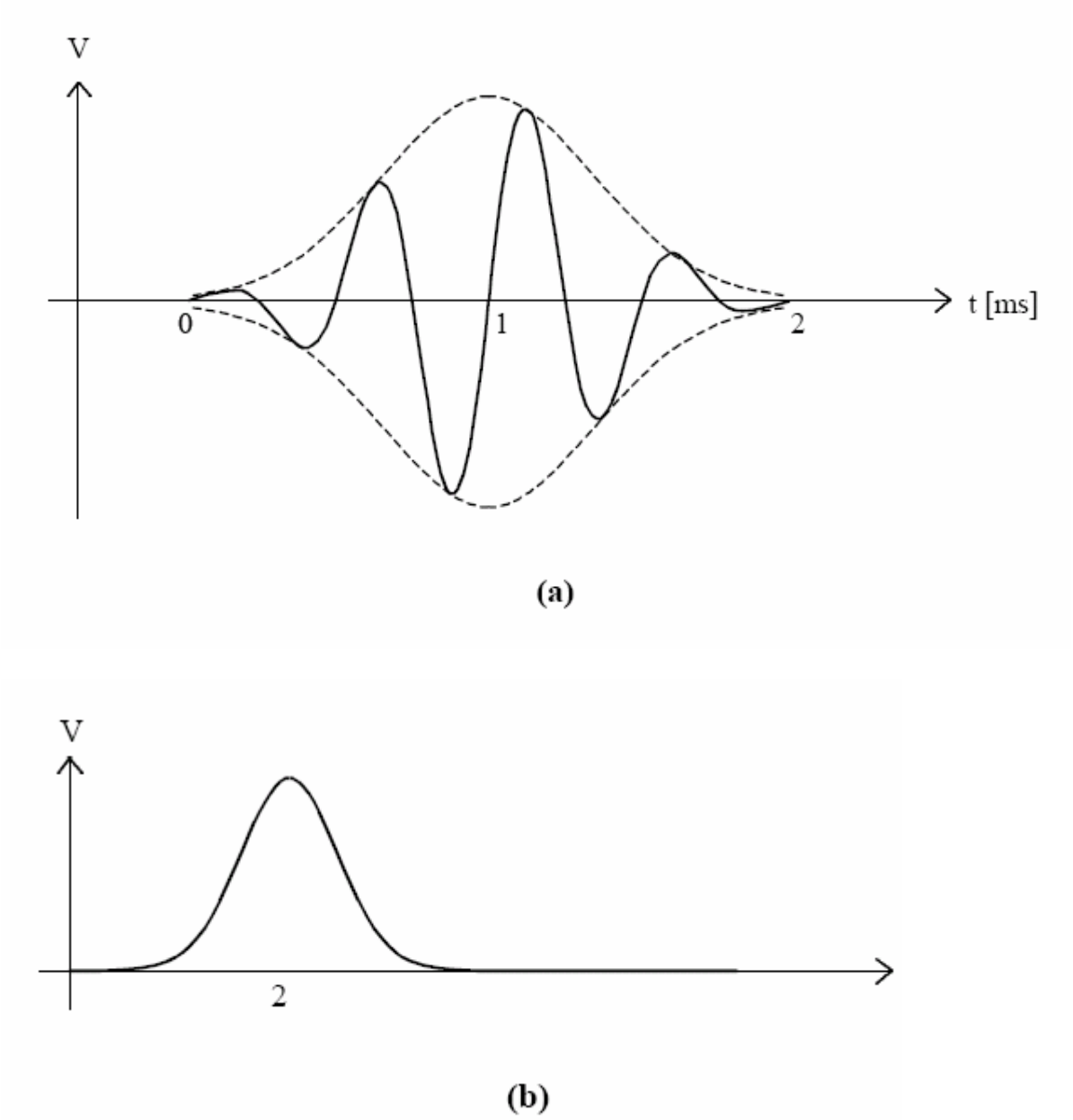

Figura 5. (a) Logon de 2ms de duración, formado por cuatro ciclos de una onda sinusoidal de 2kHz. (b) Espectro de frecuencias, que concentra más energía cerca de los 2kHz que el correspondiente al click y al tone burst.

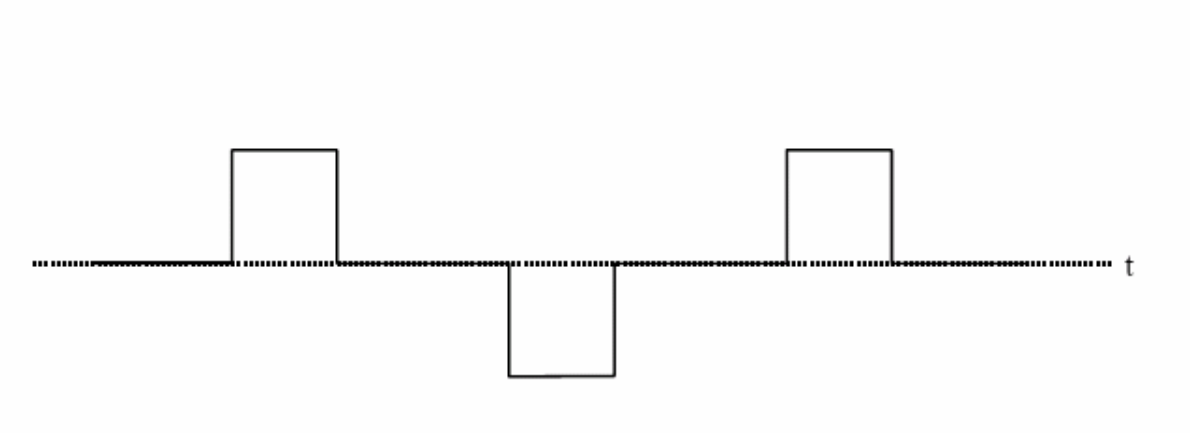

Figura 6. Click que alterna polaridades, para eliminar el artefacto de los microfónicos cocleares.

Si bien en algunos casos puede interesar medir el microfónico coclear, la mayoría de las veces constituye un artefacto que perturba la respuesta a medir. Se puede eliminar este artefacto por el simple hecho de presentar las repeticiones del estímulo alternando su polaridad (Figura 6). Al promediar las respuestas, los potenciales de acción conservarán su signo, proporcionando un promedio no nulo, mientras que los microfónicos cocleares alternarán signos, siendo su promedio 0. Esencialmente es el mismo principio que permite eliminar cualquier ruido, sólo que en este caso, como el microfónico está en sincronismo con el estímulo debe recurrirse a la alternancia de polaridades para que su promedio sea cero [13] [20].

1.7 Potencial Evocado Visual (PEV)

El PEV se produce mediante la aplicación de estímulos visuales como pueden ser un destello luminoso, o bien, un tablero de ajedrez, una palabra o una figura geométrica situados en alguna parte del campo visual. Es importante mencionar que las características del potencial son diferentes dependiendo de factores externos e internos. Se identifican como factores externos a la intensidad, tamaño del campo, frecuencia de estimulación, regularidad del patrón, color, etc. Los factores externos producen efectos diferentes sobre la forma que se obtiene en el potencial. Así mismo, los factores internos como el estado psicológico del paciente, la atención que presta, su edad, etc., también contribuyen a la forma final del potencial. Dado lo anterior, es evidente que un apropiado registro de los

potenciales depende, en gran medida, del control que se logre establecer de los factores antes mencionados [32].

1.7.1 Aspectos Fisiológicos y Anatómicos de los PEV

El cerebro procesa información en paralelo, por lo que el procesamiento de los estímulos visuales se da de esta manera. Las señales visuales son procesadas al menos por tres sistemas independientes en el cerebro. En el primero de ellos se procesa información acerca de la forma, el segundo procesa información de color, y el tercero procesa movimiento, localización y organización espacial [32].

El procesamiento de la información visual comienza cuando la luz pasa de los lentes del ojo a la retina y afecta a los foto-receptores que ahí se encuentran. En los foto-receptores, conos y bastones, se lleva a cabo una reacción fotoquímica que alimenta un proceso de transducción. La transducción tiene corno resultado la aparición de señales eléctricas conocidas como potenciales de receptor [32].

1.7.2 Aspectos Prácticos de los PEV

La retina y la córnea forman un capacitor variable el cual hace posible detectar el electrooculograma pero también produce artefactos (comportamiento anormal de la señal) en el EEG y el PEV. Con estímulos de corta duración, las células ganglionares y de la retina generan potenciales tempranos llamados ondas “a” y “b” del electro-retinograma [32].

En la retina, la parte que tiene mayor agudeza visual recibe el nombre de fóvea. En la fóvea se encuentra la mayor concentración de conos. Aquí se encuentra la mayor resolución, habilidad de reconocimiento y sensibilidad al contraste, pero poca sensibilidad al movimiento. El número de células ganglionares disminuye hacia la periferia. La visión en la fóvea es del 100 %. Es de 50 % a 2,5 grados, 33 % a 5 grados y 20 % a 10 grados del centro. De acuerdo a éstas características, el PEV se dispara centralmente por un cambio de patrón y periféricamente por luminiscencia. Los axones de las fibras de la fóvea son muy pequeños por lo que tienen una velocidad de conducción lenta comparada con la de las otras fibras ópticas. Esto causa un aumento en el tiempo de latencia del PEV cuando el estímulo se dirige únicamente a la fóvea [32].

Los componentes tempranos o primarios del PEV con latencia de 90 ms o menos se originan en el área estriada (área 17). Los componentes de alrededor de los 100 ms se originan en las

áreas 17, 18, 19 y los componentes tardíos mayores a los 130 ms, se originan probablemente en las áreas 18 y 19. Estas áreas se ilustran en la Figura 7 [32].

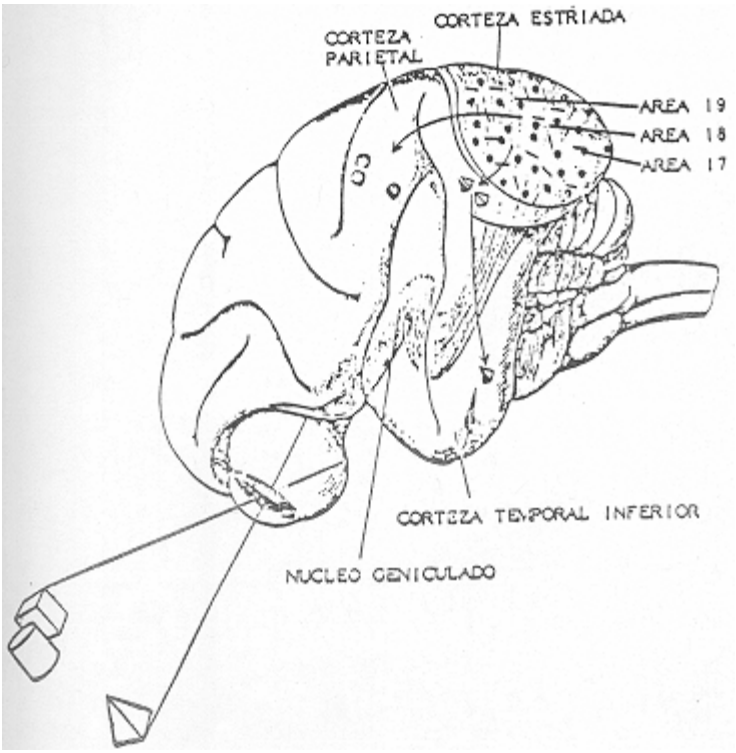

Figura 7. Áreas cerebrales asociadas a la generación de los PEV.

1.8 Algoritmos Basados en la Imagen Tiempo-Realización

Esta alternativa se basa en adquirir las respuestas neuroeléctricas y arreglarlas en una imagen 3D, que se le llama imagen tiempo-realización. A esto después pueden aplicarle técnicas de procesamiento de imágenes y demostrar que con pocas repeticiones del mismo estímulo se logra un resultado satisfactorio. Demostrando lo anterior conlleva a que el tiempo que tiene que estar el paciente cooperando para hacerse la prueba es menor. Así la prueba sería menos agotadora al paciente y la probabilidad de que se provoque diversos movimientos musculares que podrían agregar aun más ruido es menor.

Como nos damos cuenta, contar con una base de datos de imágenes tiempo-realización, es indudablemente un primer paso para un trabajo futuro, que radica en evaluar técnicas de procesamiento de imagen u otras similares que persiguen que con pocas repeticiones del estímulo se logre un resultado satisfactorio.

CAPÍTULO II: Bases para Evaluación de Algoritmos Asociados a PE

Dadas las dificultades que obtuvimos para tener señales reales del Centro de Neurociencias de Cuba que no se pudieron obtener, por eso nos movimos a otra alternativa la cual fue generar las señales reales a través de 2 artículos en PDF [17] [1] en los cuales venían señales reales en forma de imágenes. Para ello se hizo un programa en MATLAB y con el cual se obtuvieron resultados satisfactorios.

2.1 Implementación del Programa graph2vector

El programa se basa en coger píxel por píxel o sea ir barriendo la imagen y evaluando el valor del píxel (que corresponde al color que lleva) y como la imagen es de blanco y negro, y sabiendo que el color blanco en el MATLAB tiene como valor 255 entonces, una solución pudiera ser tomar solo los valores que son por debajo de 50 y así logramos tener solo los puntos negros de la imagen y tener la señal real. También hay que decir que antes de trabajar sobre la imagen trabajamos con el programa de Windows XP que es el PAINT y se borraron las letras como P1, P2, N1, N2, etc. y se dejó la señal como tal. También dentro del programa nos damos cuenta que se tuvo que escalar la señal porque cuando obtuvimos los valores de los píxeles negros, estos nos dan idea de la forma de la señal o sea la posición del píxel pero no se corresponden a los valores exactos de la señal, por eso después de obtener la forma de onda de la señal se escaló la señal tanto en el eje de tiempo como en amplitud. Para escalar el eje de tiempo se usó la función *linspace* a la cual se le pasa los valores de Tmin y Tmax (son los valores de tiempo que usted quiere que tenga la señal) y también la cantidad de muestras de la señal. Señalamos que a la hora de escalar el eje de tiempo se pudo haber diezmado la señal y en este caso da resultado o sea no se pierde la forma de la señal porque en primera el tamaño de la imagen es grande y se pudo sacar muchas muestras y en segunda la señal no tiene muchos picos determinantes y eso facilita el trabajo, o sea se puede diezmar con un valor que es (el número de muestra de la señal original / el intervalo requerido) y este valor quiere decir si es 8 por ejemplo que de cada 8 muestras el toma una, por eso cuando la señal tiene muchas muestras y no tiene muchos picos determinantes se puede aplicar esta metodología.

2.1.1 Simulación del VEP de Patrón Invertido

El VEP de patrón invertido tiene variaciones relativamente bajas en cuanto a la forma de onda y latencia de pico en toda la población. El VEP de patrón invertido consta de picos en N75, P100 y N135. La nomenclatura consiste para definir los picos como picos negativos o picos positivos y los números después de la letra indican el tiempo en que ocurre el pico. Es recomendable medir la latencia o sea el tiempo entre N75 y P100 y este tiempo está afectado por algunos factores como el tamaño de la pantalla, el contraste, promedio de la luminancia, error refractivo, pobre fijación y miosis.

Luego se muestra la respuesta neuroeléctrica para ese tipo de potenciales evocados visuales en la Figura 8.

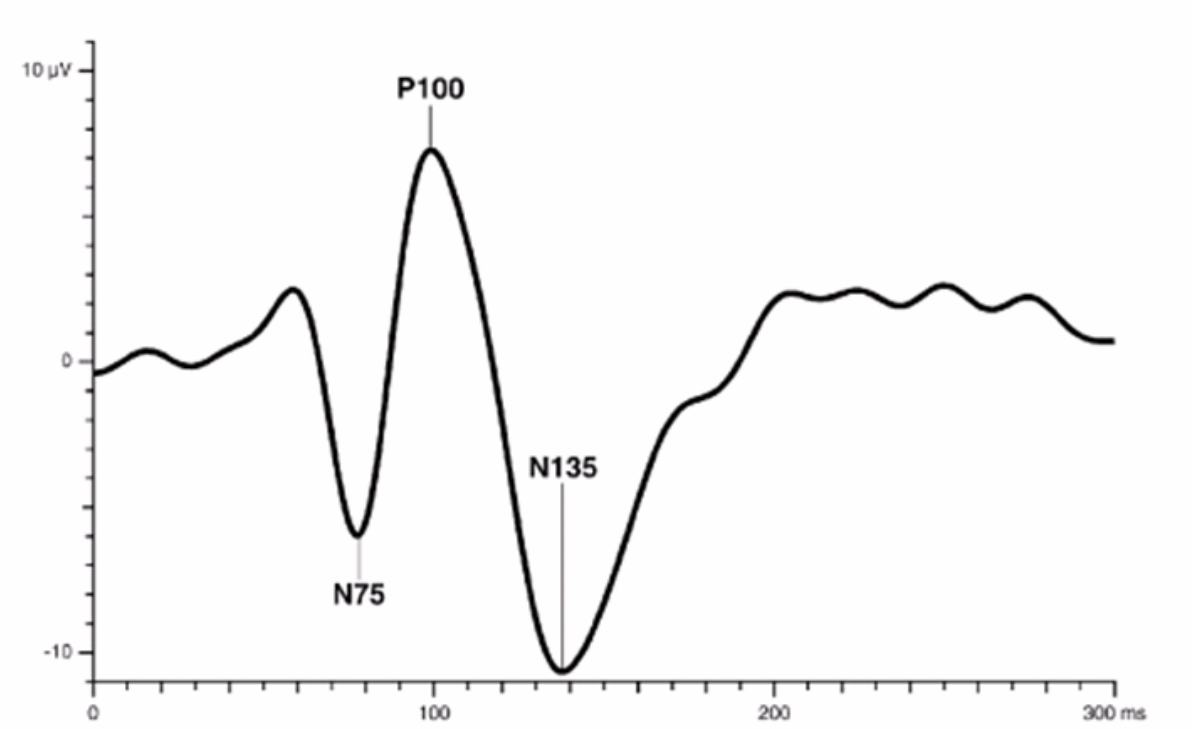

Figura 8. Respuesta neuroeléctrica para el VEP de patrón invertido.

Luego se muestra el resultado que obtuvimos en la Figura 9.

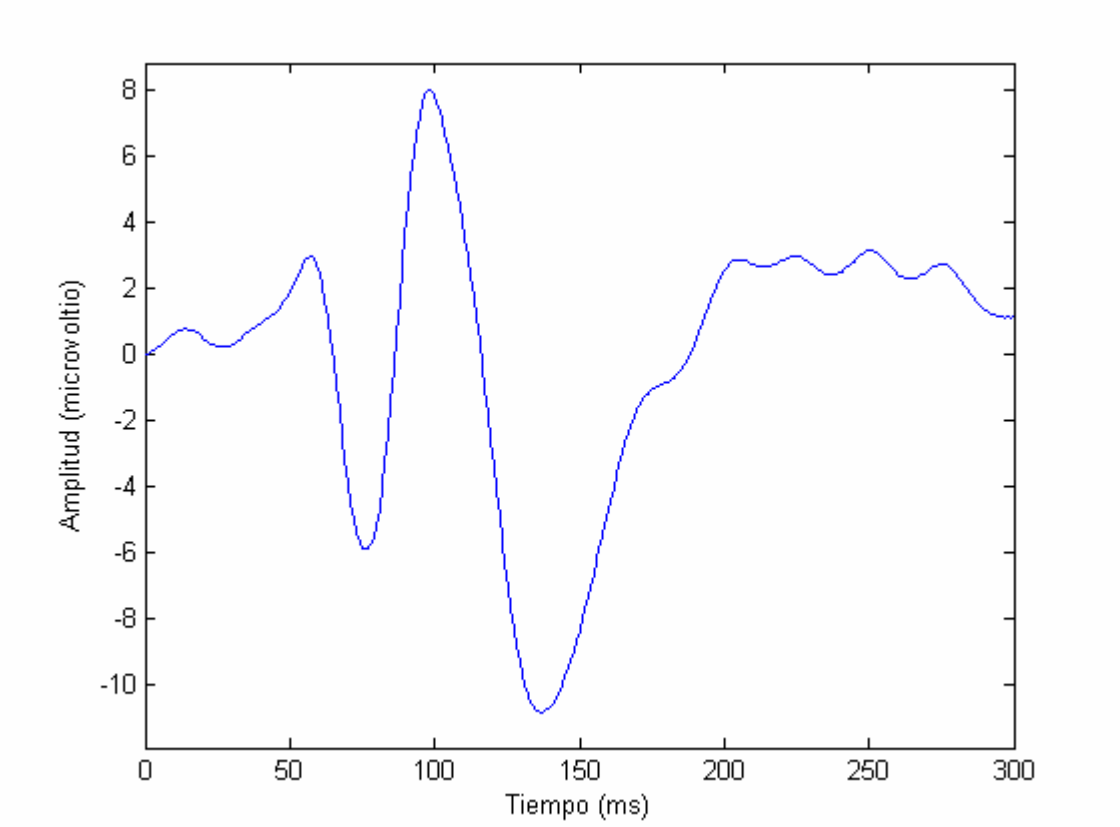

Figura 9. Resultado obtenido a través del programa graph2vector.

2.1.2 Simulación del VEP Tipo Flash

El VEP tipo flash (Figura 10) es mucho más flexible que el de respuesta patrón, es usado para los pacientes que no pueden cooperar con los VEP de patrón. Los potenciales evocados visuales a través de flash consisten de un conjunto de picos tanto positivos como negativos. El pico más temprano ocurre aproximadamente a los 30 ms. Para ese tipo de potenciales evocados los picos más determinantes son los picos P2 y N2.

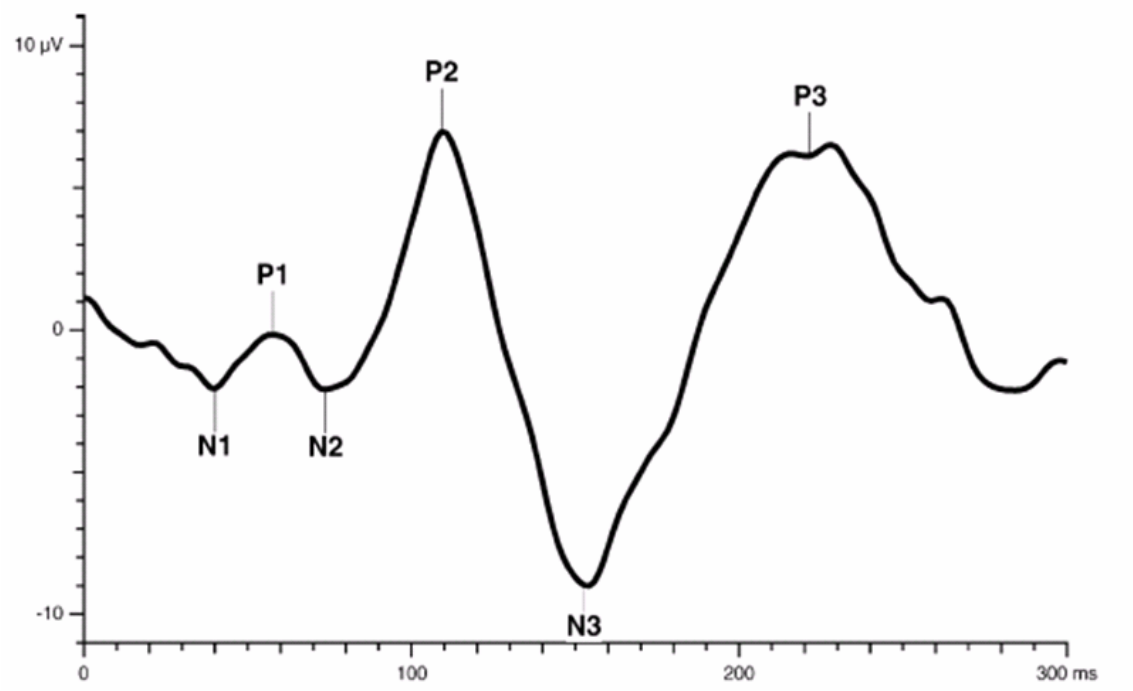

Figura 10. VEP tipo flash.

Luego se muestra el resultado obtenido en la Figura 11.

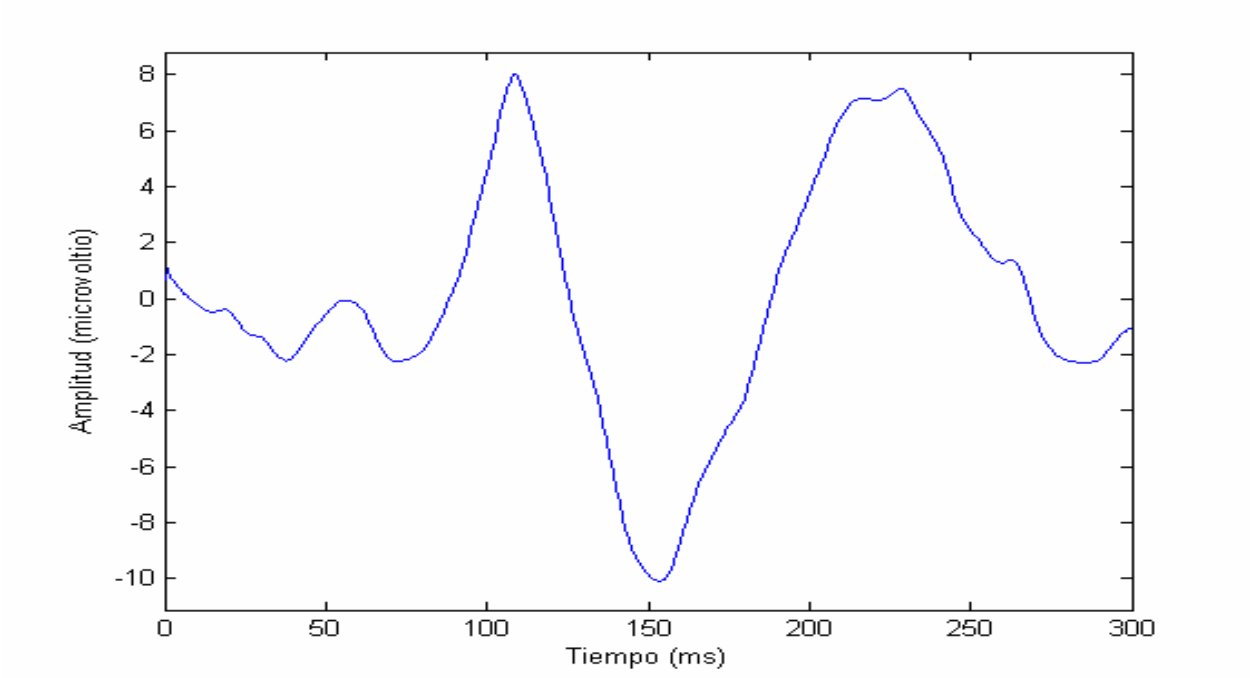

Figura 11. Resultado obtenido para el VEP tipo flash.

2.1.3 Simulación del VEP onset/offset

El VEP *onset /offset* (Figura 11) es más flexible aparentemente que el VEP de patrón invertido. La respuesta del estímulo típicamente consiste de 3 picos fundamentales en adultos: C1 (pico positivo que ocurre a los 75 ms aproximadamente), C2 (pico negativo que

ocurre a los 125 ms aproximadamente) y C3 (pico positivo que ocurre aproximadamente a los 150 ms). El resultado obtenido se muestra en la Figura 12.

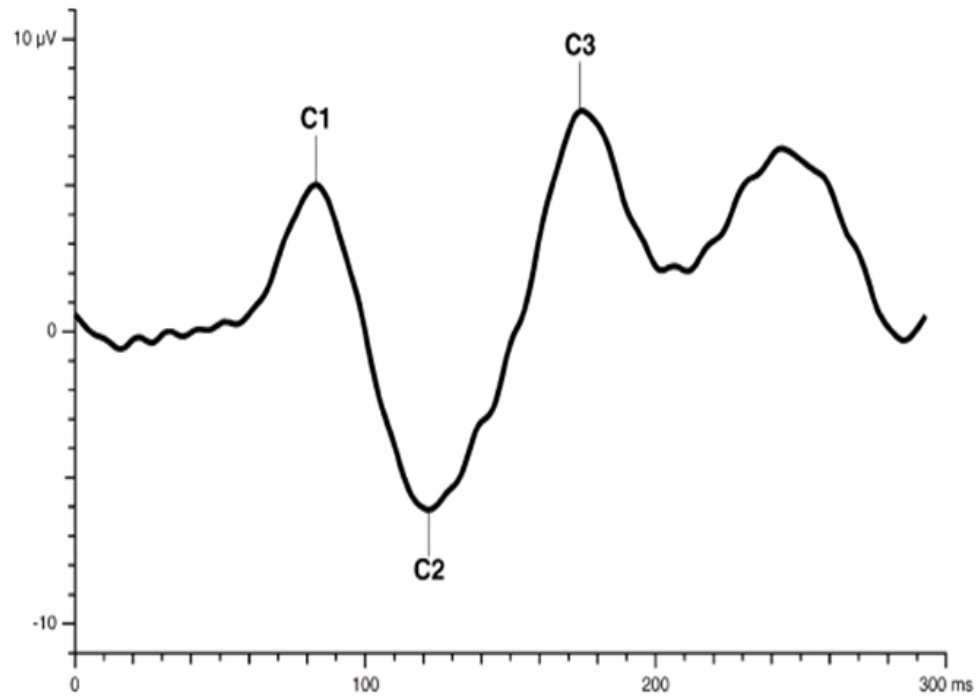

Figura 11. Respuesta neuroeléctrica para el VEP *onset /offset.*

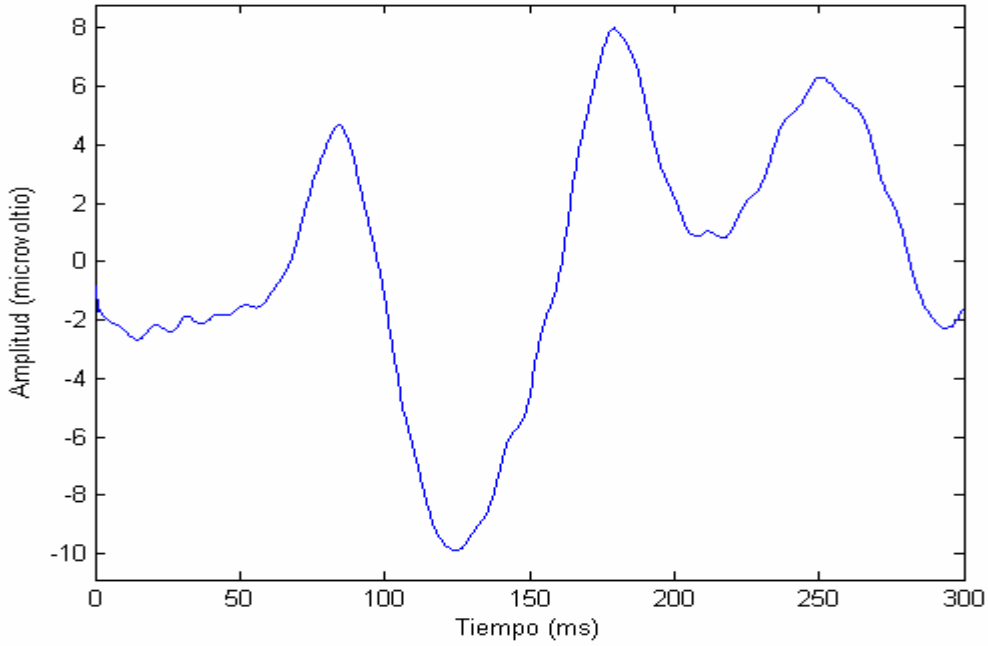

Figura 12. Resultado obtenido a través del programa graph2vector.

Como nos damos cuenta de los resultados que las señales son idénticas, señalamos que la señal generada esta en escala tanto en el eje de tiempo como en magnitud.

2.1.4 Simulación de Potenciales Evocados Auditivos de Tronco Cerebral

También escogimos una señal PEATC (Potenciales evocados auditivos de tronco cerebral) y le aplicamos el mismo el algoritmo pero ante todo daremos una explicación teórica de lo que es el PEATC.

Los PEATC son señales eléctricas que representan la respuesta a una estimulación repetitiva de las fibras del nervio auditivo y de las vías auditivas ascendentes dentro del tronco cerebral. La morfología característica en adultos se muestra en la Fig. 13, donde se puede observar la secuencia de picos, que se identifican con números romanos desde el I hasta el VII. De éstos, los más importantes son los picos I, III y V [1].

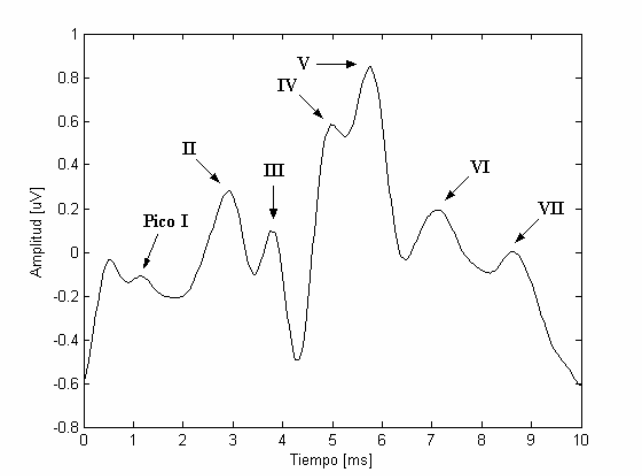

Fig. 13. Morfología de la señal de PEATC.

En la Fig. 14 se muestra el espectro de frecuencias de la señal de PEATC. El mismo tiene como características que la energía de la señal está por debajo de los 1,5 kHz y que se concentra en tres regiones del espectro, una región de frecuencia baja alrededor de los 100 Hz, otra de frecuencia media alrededor de los 500 Hz y finalmente una de frecuencia alta alrededor de los 1000 Hz [1].

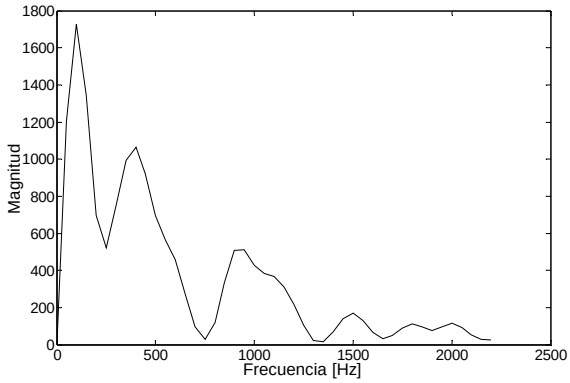

Fig. 14. Espectro de frecuencias de la señal de PEATC.

Cada una de estas regiones aporta energía a determinados picos de la señal. La energía de los picos I y III corresponde en parte a la región de frecuencia media y en parte a la de frecuencia alta. La mayor parte de la energía del pico II corresponde a la componente de frecuencia alta y la energía del pico IV corresponde a la componente de frecuencia media, al igual que la mayor parte de la energía del pico V [1].

Se uso el mismo algoritmo para obtener la señal real y ahí obtuvimos resultado satisfecho. Señalamos que se tuvo que escalar la señal con la función *linspace* también pasándole los mismos parámetros puestos arriba y decimos que en este caso como una alternativa que uno pudiera pensar para escalar seria diezmar pero en este caso se nos hace difícil porque la señal tiene muchos picos determinantes, entonces a la hora de diezmar se pierde la forma porque la cantidad de muestras son pocas (porque el tamaño de la imagen es pequeño), no como en los primeros casos que pudimos sacar muchas muestras (píxeles) entonces cuando diezmamos no se perdió la forma de señal sino obtuvimos una señal perfecta.

Luego se muestra el resultado en Figura 15.

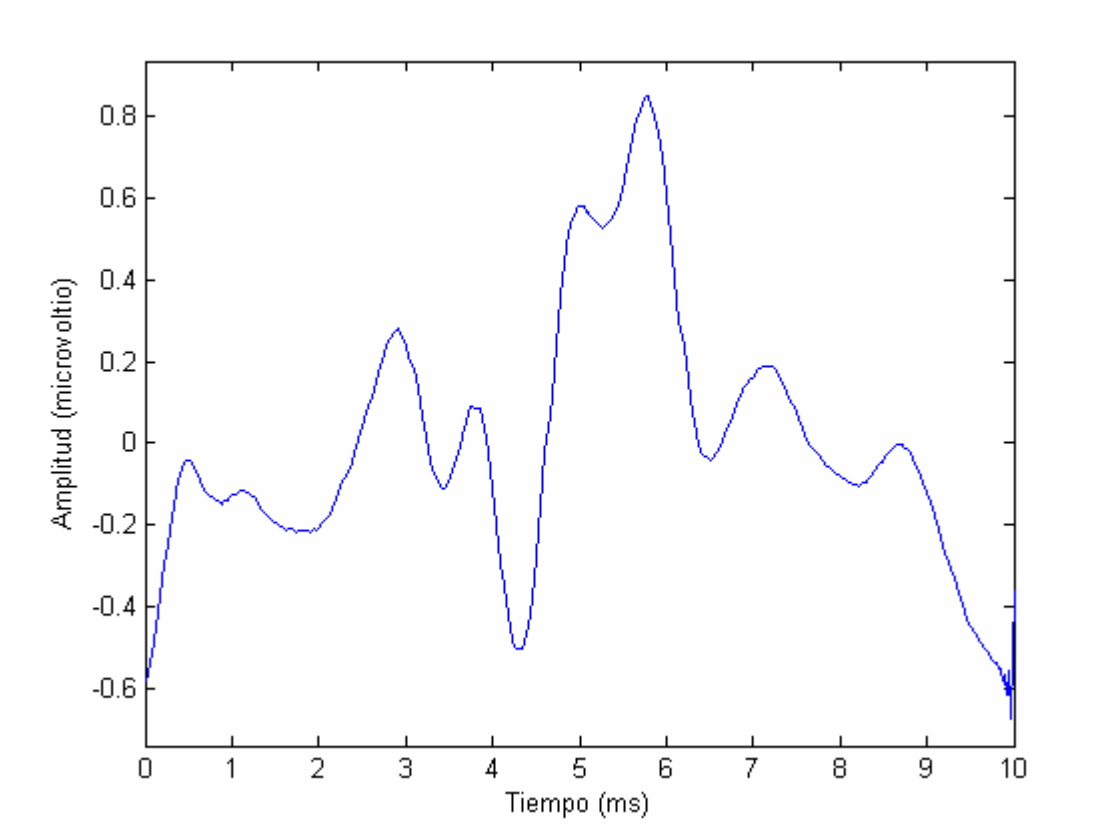

Figura 15. Resultado obtenido a través del programa graph2vector.

2.2 Implementación del Programa plotvector

También se hizo otro programa que se llama *plotvector* en que se le entra como datos el vector de muestras tomadas y la frecuencia de muestreo y el lo que hace es dibujar la señal. El programa lo que hace primero es sacar cuantas muestras se han tomado y después multiplica la cantidad de muestras por la frecuencia de muestreo y así tendrá el tiempo total de la prueba. Sabiendo el tiempo total de la prueba y sabiendo que entre una muestra y otra la diferencia es el tiempo de muestreo que es el inverso de la frecuencia de muestreo y teniendo los valores de cada muestra ya se puede recuperar la señal mediante la función **spline** y dibujarla mediante la función **plot**.

2.3 Metodología para Mejorar la Relación Señal a Ruido

Ahora se propone de un método para mejorar la relación señal a ruido.

El esquema del método de mejoramiento de la RSR propuesto se muestra en la Figura 16.

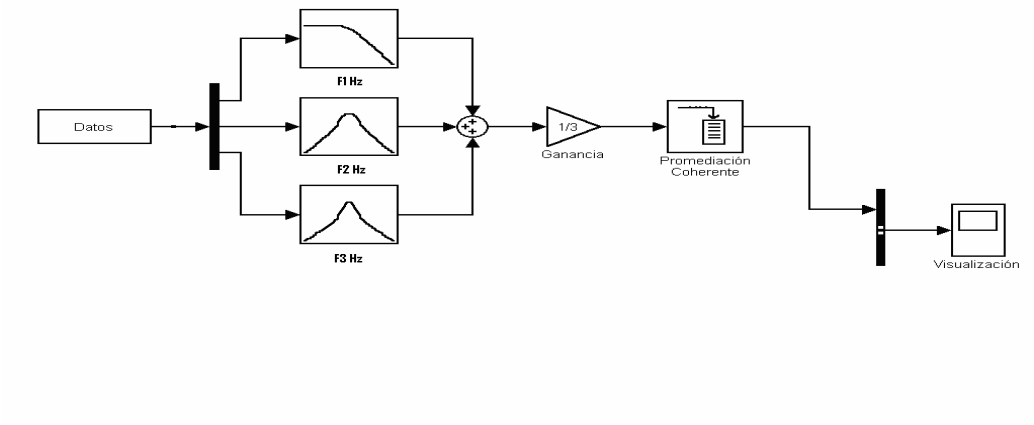

Figura 16. El esquema propuesto para mejorar la relación señal a ruido.

Cada época registrada se pasa por un banco de filtros que enfatiza el espectro de frecuencias de los PEV. La salida de cada filtro es sumada para conformar la época filtrada y posteriormente se la promedia para extraer la señal de interés. El objetivo es que cada una de las épocas filtradas tenga una mejor RSR que las originales. En consecuencia, se necesitarán menos épocas para obtener la señal de PEV.

Para la implementación de los filtros se tuvo en cuenta la distribución del espectro de frecuencias de los PEV. Considerando esto, más el aporte de cada región del espectro a la energía de los picos, fue posible realizar un filtrado de la señal que atenúe o elimine las componentes que no son de interés para el análisis. Se utilizaron filtros pasa bajos tipo Bessel, Butterworth y Chebyshev tipo 1 y tipo2, de orden 1 y 2, con frecuencias de corte o centrales en 20 Hz para comparar el desempeño de los mismos.

Para no alterar la fase de las señales de PEV, se realizó un filtrado digital de *fase cero*, y eso en MATLAB7.0 se logra usando *filtfilt* en vez de *filter*, procesando primero la entrada en la dirección temporal para luego invertir la secuencia y filtrarla nuevamente. Esto es importante ya que los valores de latencia de los picos son los parámetros más importantes en el diagnóstico de patologías, y se ven afectados al variar la fase de la señal.

En este caso como el espectro de frecuencia de PEV esta concentrado en una zona única de alrededor de 20 Hz (viendo el espectro de frecuencia de la señal en la Figura 32) pues solo hizo falta un filtro en este caso y no un banco de filtros. En este caso especial lo que se hizo fue coger la señal con ruido y pasarla por el filtro y luego verla pero a veces viendo la señal uno a simple vista no se da cuenta de que si se mejoró o no la RSR, por eso es recomendable

calcular la relación señal ruido y así comparando los valores nos damos cuenta de que si se mejoró la RSR o se empeoró.

A la señal de PEV sacada del artículo se le adicionó una interferencia a 60 Hz (que siempre afecta ese tipo de señales) y una interferencia a 180Hz y cabe señalar que se adicionó el ruido a través del programa *AVETESTNOISE*.

La señal que se tomó para el análisis fue el VEP tipo flash y se muestra en la Figura 17.

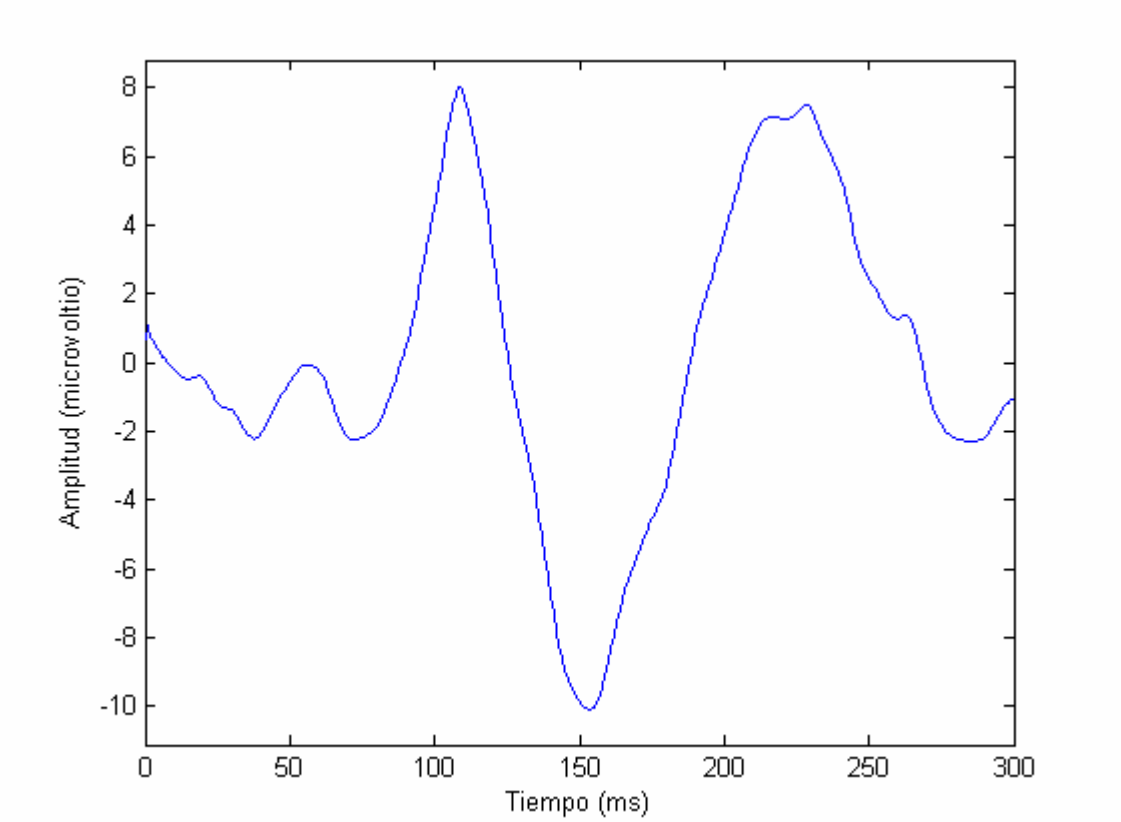

Figura 17. VEP tipo flash que se tomó para el análisis.

Luego se muestra la señal original (Figura 19), el ruido solo (Figura 18) y las 2 señales compuestas en la Figura 20.

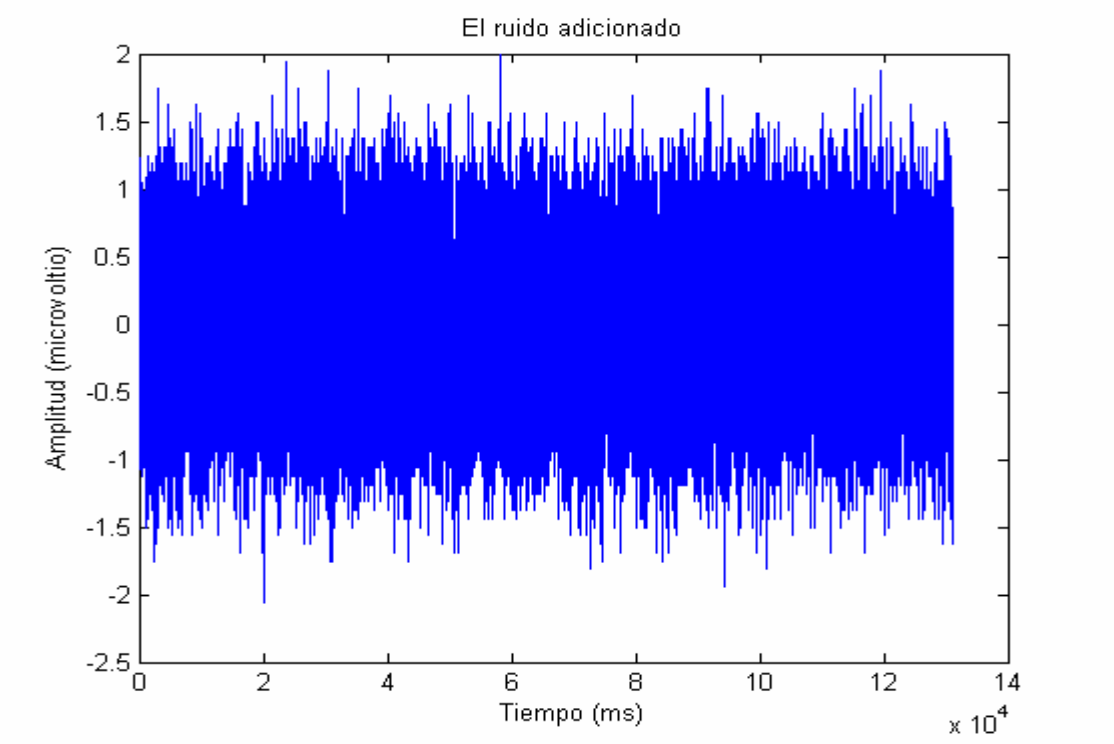

Figura 18. Ruido adicionado a la señal de VEP.

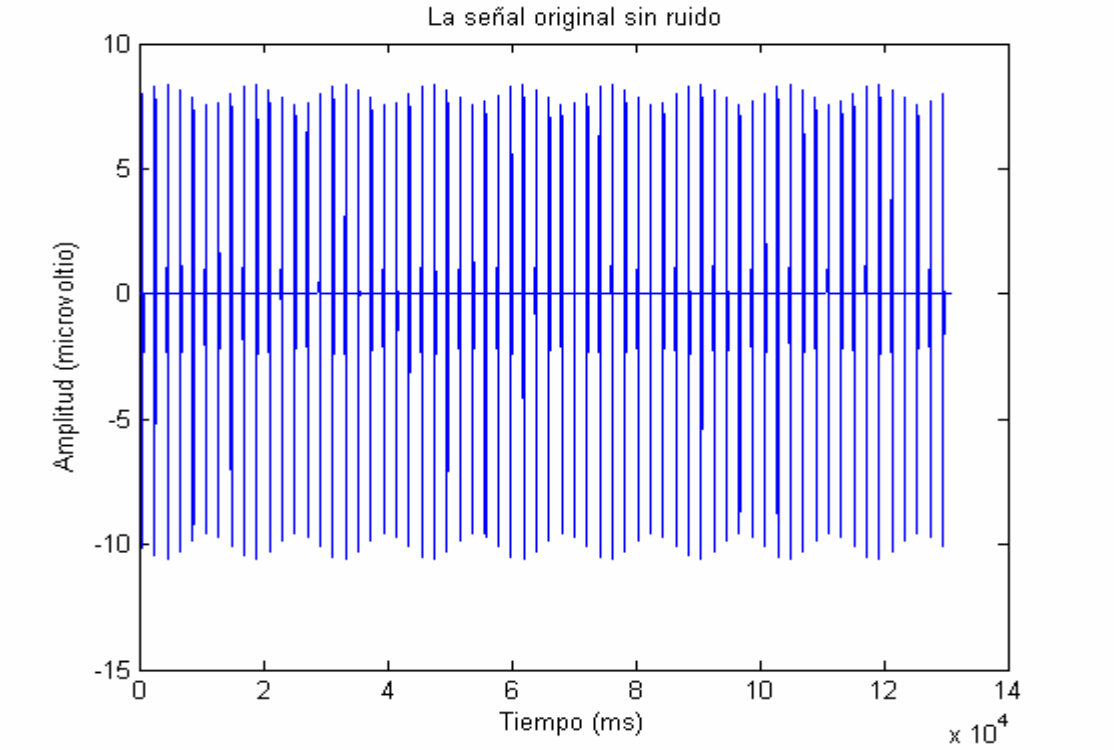

Figura 19. La señal original sin ruido.

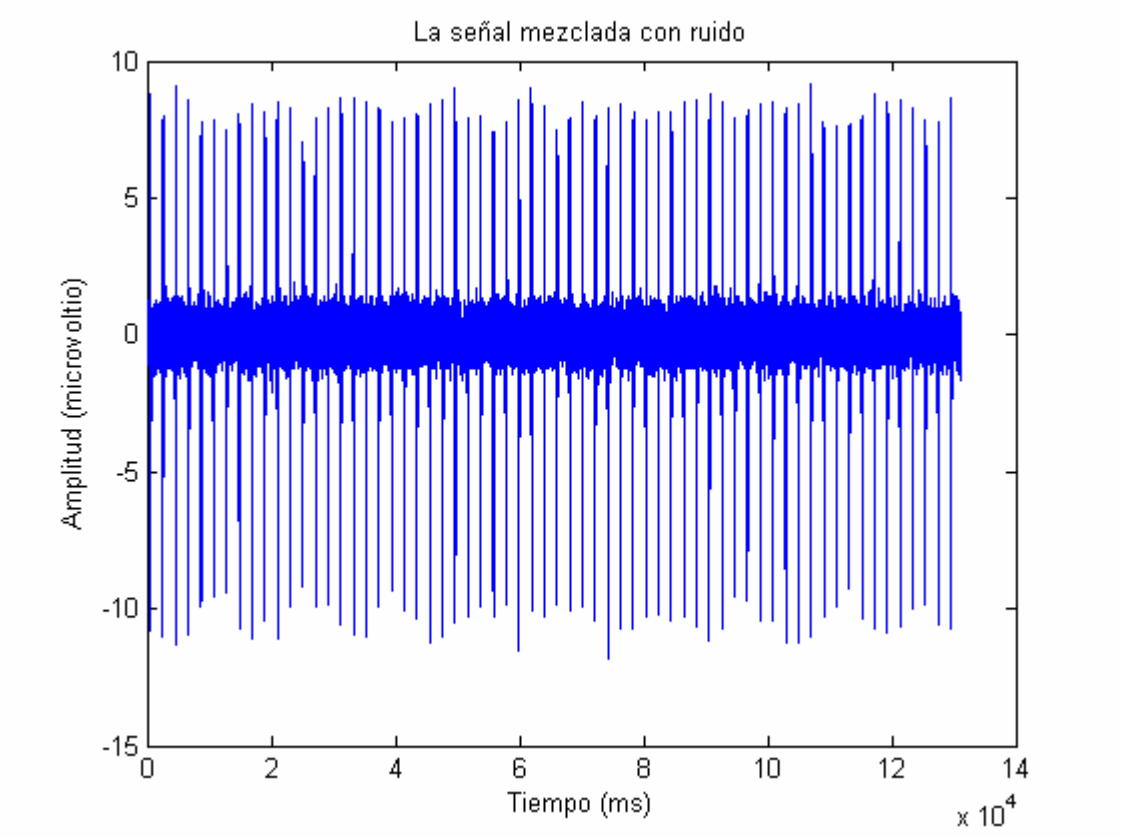

Figura 20. La señal original junto con el ruido adicionado.

Se pasó la señal con ruido por filtros paso bajo de tipo Bessel, Butterworth, Chebychev tipo 1 y tipo 2, de orden 1 y 2, y se calculó la relación señal ruido de cada caso.

Antes de todo se explicara como se calcula la relación señal a ruido de la señal.

Hay que tener en cuenta primero que la varianza es aproximadamente la potencia de la señal y por eso teniendo la señal sin ruido se calculó la potencia de la señal de esta forma **spotencia =var(signal);**

Ahora después de adicionar el ruido, se restó la señal con ruido de la señal sin ruido y así nos quedamos con el ruido y calculamos la potencia del ruido.

noise= signalwnoise-signal;

npotencia=var(noise);

Entonces la RSR (relación señal a ruido) es **RSR=10*log(spotencia/npotencia);**

Ahora para calcular la RSR de cualquier señal filtrada lo que tenemos que hacer es restar la señal filtrada de la señal original y así nos quedamos con el ruido y calculamos la potencia de ruido.

fnoise= fsignalwnoise-signal;

nfpotencia= var(fnoise);

Luego se calcula la RSRF (relación señal a ruido de la señal filtrada) de esta forma

RSR=10*log(spotencia/nfpotencia);

La RSR de la señal original nos dio 24,6784 dB.

1- Como primer filtro se tomó el *Bessel* de orden 20 y una frecuencia de corte de 0,04, esta frecuencia se calcula como (frecuencia de corte requerida en Hz /mitad de frecuencia de muestreo) y en este caso la frecuencia de muestreo es 1 kHz y la frecuencia de corte requerida es de 20 Hz porque como vimos en el espectro de frecuencia de la señal que hay una zona donde se concentra la mayor parte de la energía y es alrededor de 20 Hz.

Luego se muestra la señal filtrada en la Figura 21.

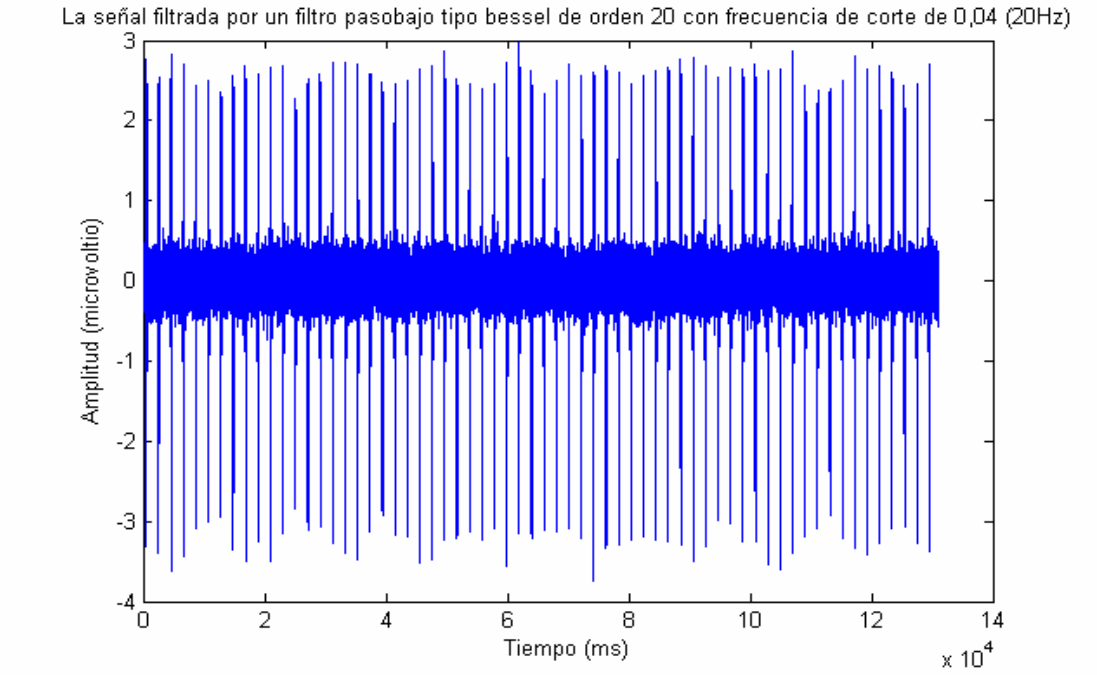

Figura 21. Señal filtrada por un filtro de tipo Bessel.

La RSR de la señal filtrada nos dio 7,0682.

2- Se filtró la señal original con un filtro paso bajo *Butterworth* de orden 2 y frecuencia de corte de 0,04 (20 Hz). Luego se muestra la señal filtrada en la Figura 22.

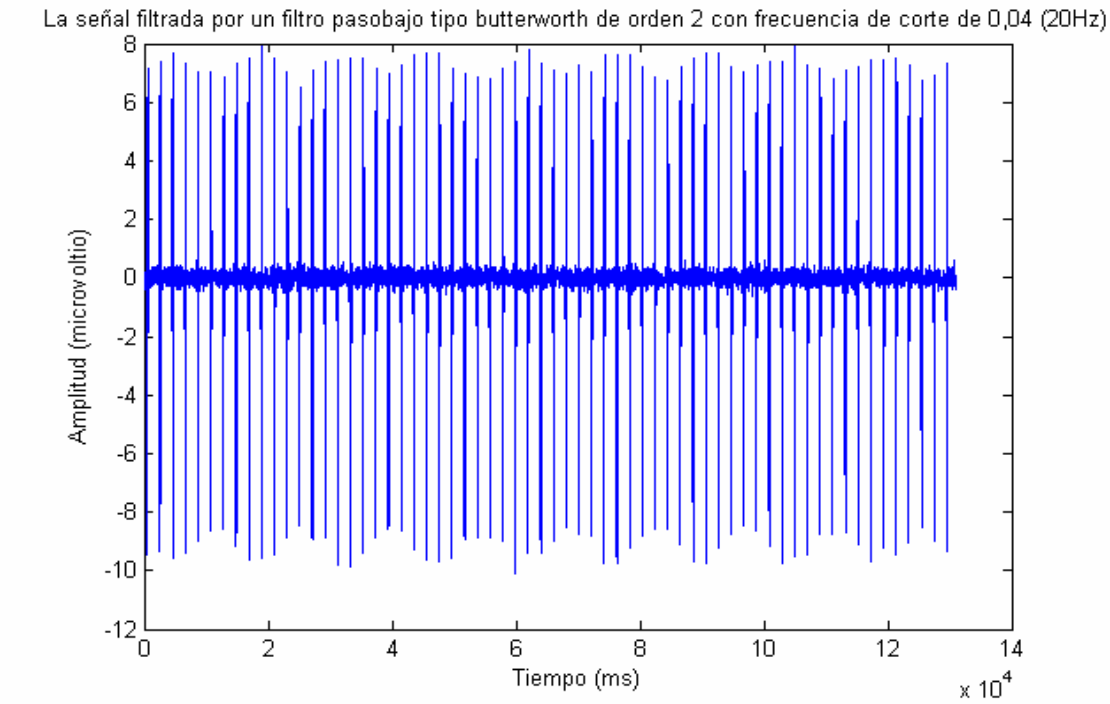

Figura 22. Señal luego de pasar por el filtro mencionado anteriormente.

La RSR de la señal filtrada es 35,7381dB.

3- Se pasó la señal original por un filtro paso bajo de Butterworth de primer orden con una frecuencia de corte de 0,04 (20 Hz) y luego se muestra la señal filtrada en la Figura 23.

La RSR de la señal filtrada es 30,4958dB.

4- Se pasó la señal original por un filtro paso bajo de Chebychev tipo 1 de primer orden con una frecuencia de corte de 0,04 (20 Hz) y luego se muestra la señal filtrada en la Figura 24. La RSR de la señal filtrada es 0,1998dB.

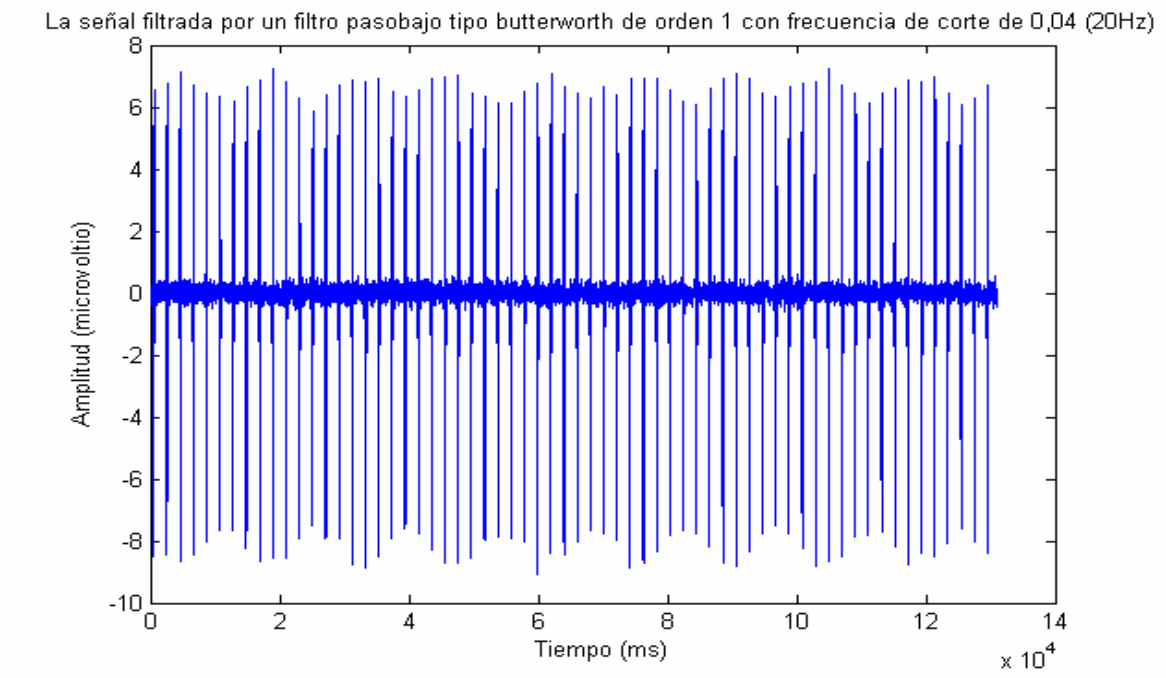

Figura 23. Señal filtrada por el filtro mencionado anteriormente.

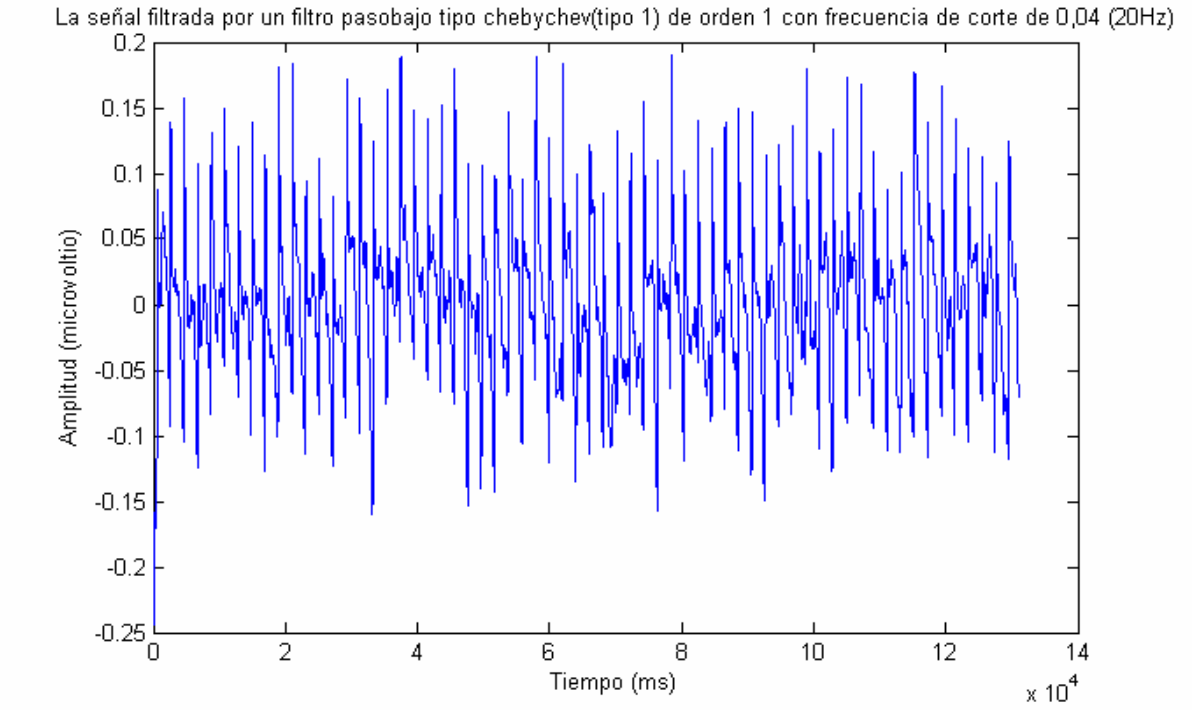

Figura 24. Señal filtrada por el filtro mencionado anteriormente.

5- Se pasó la señal original por un filtro paso bajo de *Chebychev* tipo 2 de primer orden con una frecuencia de corte de 0,04 (20 Hz) y luego se muestra la señal filtrada en la Figura 25. La RSR de la señal filtrada es 0,1998dB.

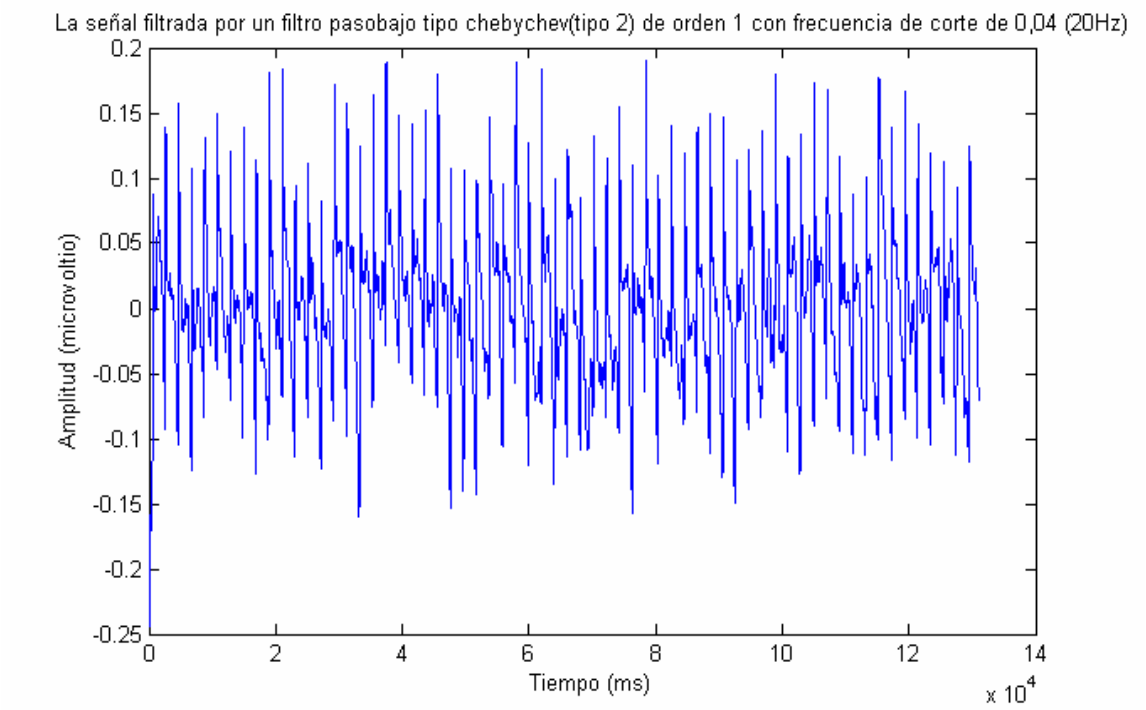

Figura 25. Señal filtrada por el filtro mencionado anteriormente.

6- Se pasó la señal original por un filtro paso bajo de *Chebychev* tipo 1 de segundo orden con una frecuencia de corte de 0,04 (20 Hz) y luego se muestra la señal filtrada en la Figura 26.

La RSR de la señal filtrada es 0,1844 dB.

7- Se pasó la señal original por un filtro paso bajo de *Chebychev* tipo 2 de segundo orden con una frecuencia de corte de 0,04 (20 Hz) y luego se muestra la señal filtrada en la Figura 27. La RSR de la señal filtrada es 3,5702dB.

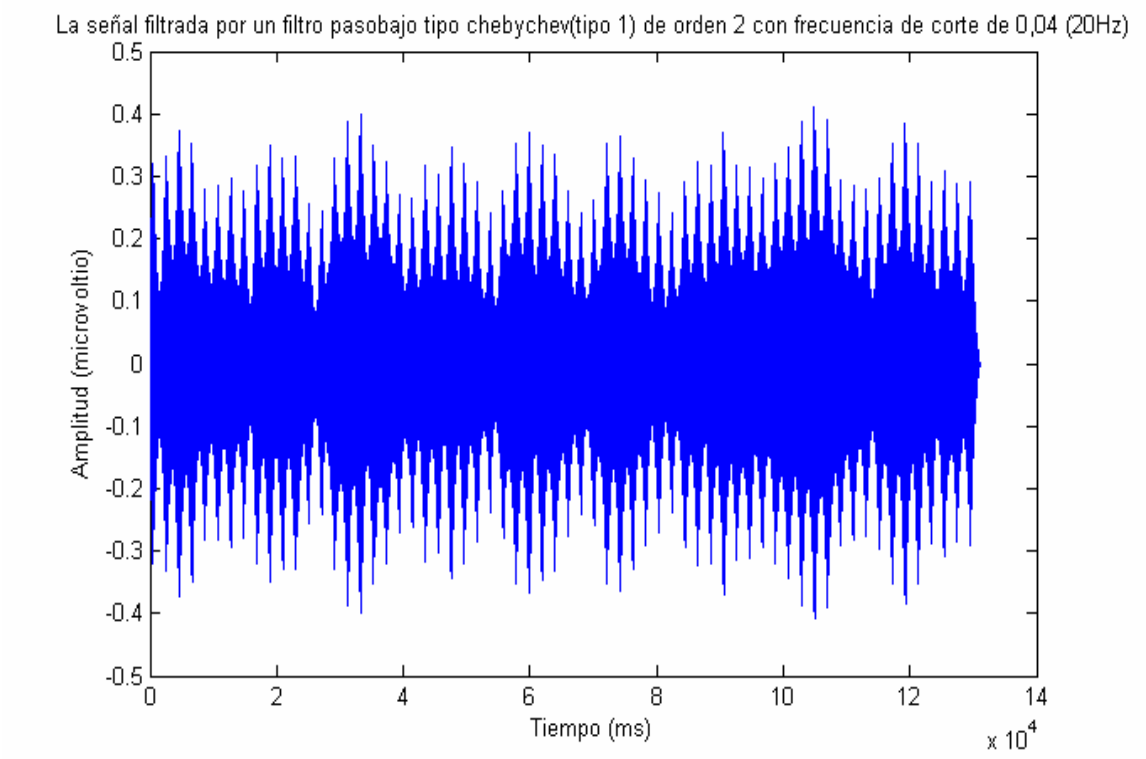

Figura 26. Señal filtrada por el filtro mencionado anteriormente.

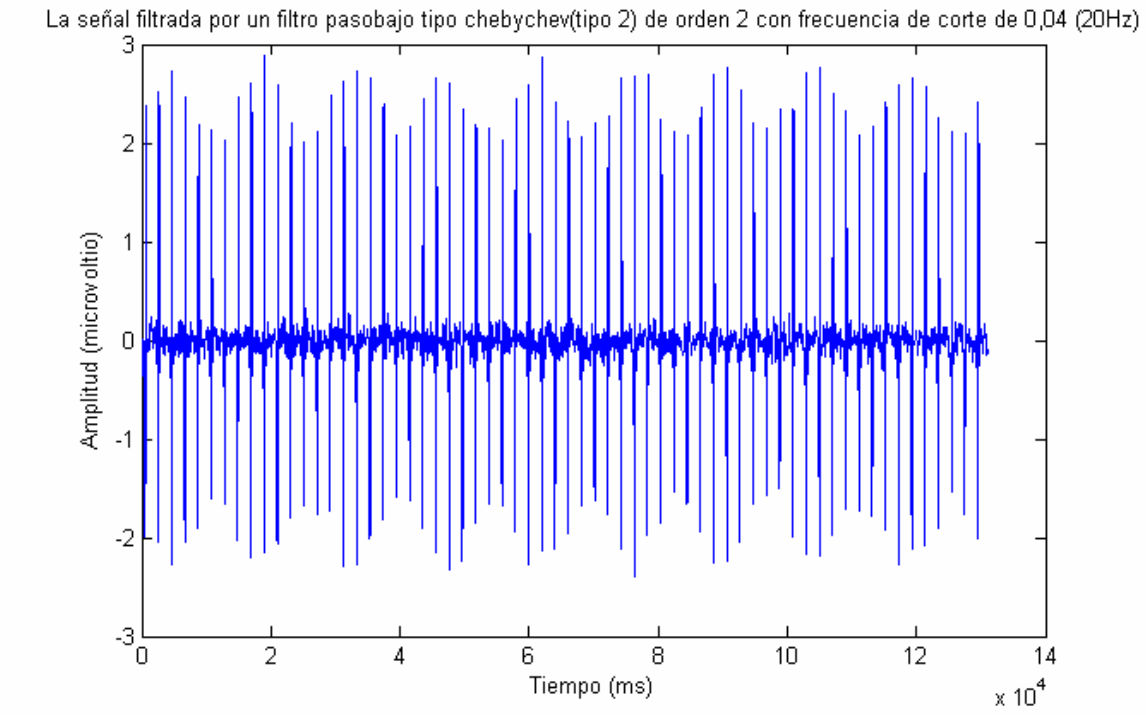

Figura 27. Señal filtrada por el filtro mencionado anteriormente.

Luego se hizo una gráfica de los resultados y se demostró que el filtro de Butterworth de segundo orden es el mejor filtro para ese tipo de señal (Figura 28).

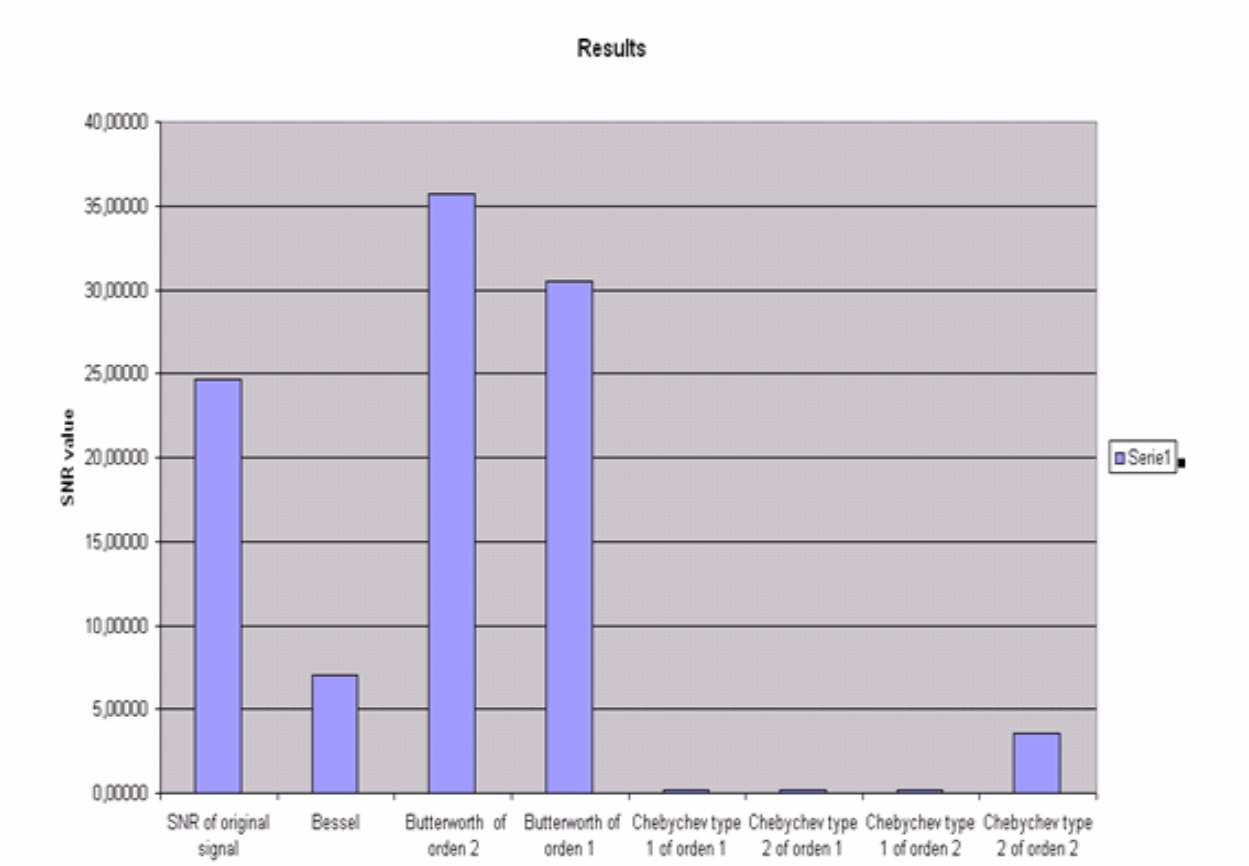

Figura 28. La figura obtenida en Excel que muestra el valor de RSR en dB para cada filtro usado.

CAPÍTULO III: Resultados de Simulaciones de PEV

3.1 Simulaciones de PEV

Después de obtener la señal real se utilizó un programa el cual se llama AVETESTSIGNAL y lo que hace es coger las respuestas neuroeléctrica y ponerlas en una imagen 3D a la cual se le llama imagen tiempo-realización y como se espera para un trabajo del futuro que sería demostrar de la base de datos de imágenes tiempo-realización aplicándole técnicas de procesamiento de imagen que con pocas repeticiones se logra resultados satisfactorios.

Después de obtener la señales reales como vimos en el capitulo 2 pues se usaran 2 programas el cual uno se llama AVETTESTSIGNAL y otro que se llama AVETESTNOISE.

Este programa (AVETESTSIGNAL) genera *event-related signals in a wide sense* (ERSWS) para estudiar señales 'promediadas'. Las señales son generadas de la onda básica dándole el tipo de *shimmer* (modulación de amplitud), *jitter* (desplazamiento relativo de las mismas) y *width modulation* (modulación de ancho) que afectan la señal básica en cada respuesta.

Señalamos aquí y que es muy importante recordarlo que la modulación de amplitud se debe en la practica a la concentración del paciente o sea mientras más concentrado el paciente esta, la amplitud es máxima pero mientras más dure la sesión el paciente se cansa y empieza a perder la concentración y también disminuye la amplitud, esa variación de amplitud es lo que decimos modulación de amplitud y también que la latencia expresa velocidad de procesamiento de información y eso indica que la latencia depende mucho de la edad porque de lógico que la latencia de un adulto es mucho más rápido de un anciano porque un anciano procesa más lento la información que un adulto.

La segunda función la cual es AVETESTNOISE lo que hace generar ruido para ERSWS para estudiar señales 'promediadas'. Ruido e interferencias son generadas para unas determinadas amplitudes, frecuencias y fases requeridas.

Empezando por la primera señal de potencial evocado visual normal tipo flash, usando la función AVETESTSIGNAL se obtuvieron los resultados mostrados a continuación.

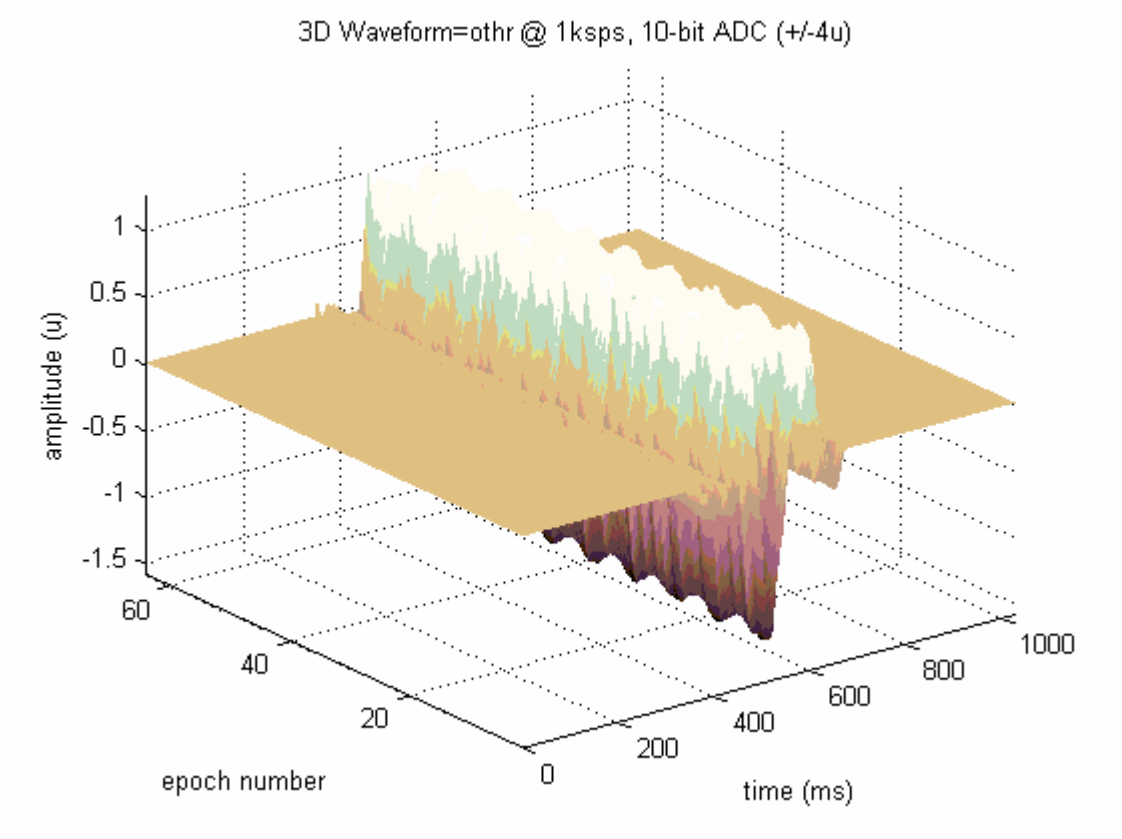

Figura 29. La imagen tiempo-realización obtenida a través del programa avetestsignal.

En la Figura 29 lo que se hizo fue coger las respuestas neuroeléctrica y ponerlas en una imagen 3D a la cual se le llama imagen tiempo-realización teniendo en cuenta las variaciones de *jitter*, *shimmer* y *width modulation* que fueron para esta prueba. En cuanto al *shimmer* (modulación de amplitud) fue una variación sinusoidal y para el *width modulation* fue también sinusoidal y para el *trigger* fue aleatorio. También nos damos cuenta que las coordenadas de la imagen son el tiempo, el número de la respuesta neuroeléctrica (que en este caso son 64) y la amplitud que en este caso es normalizada.

Señalamos que el AVETESTSIGNAL también se toma en cuenta la cantidad de bits del convertidor analógico digital que determina la resolución, como vemos que se tomo el valor por defecto que es 10 en este caso.

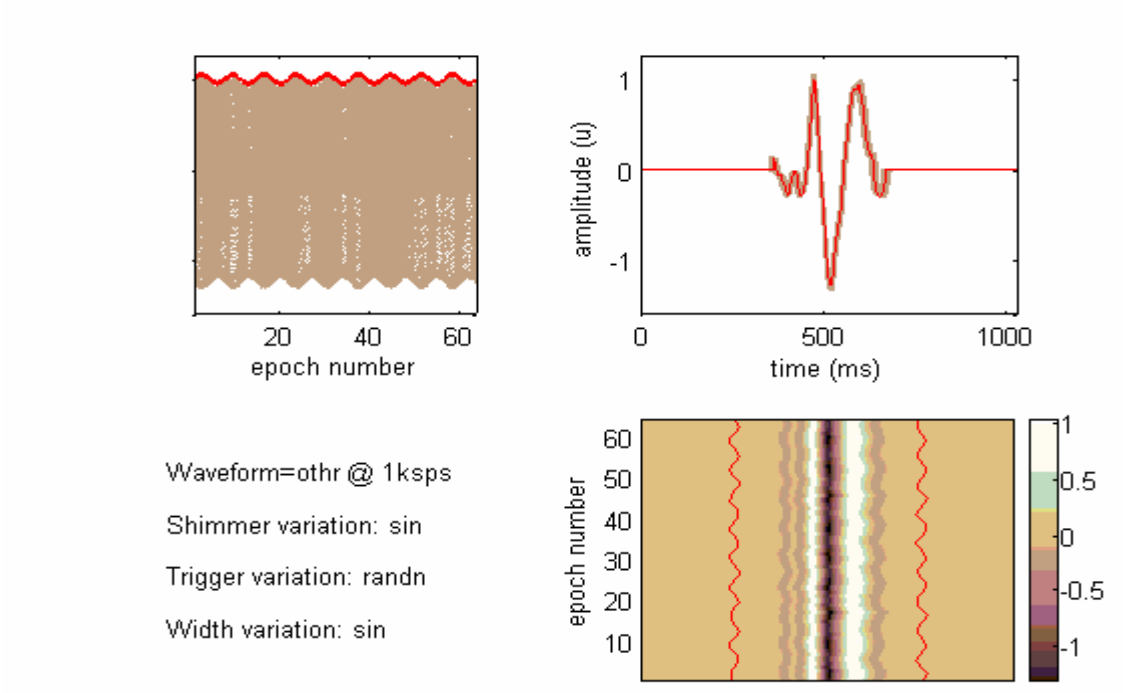

Figura 30. Las proyecciones de la imagen tiempo-realización en los 3 planos.

En la Figura 30 se muestra las proyecciones de la imagen tiempo-realización en los 3 planos, XY, XZ, y YZ.

En la primera se muestra la proyección de la imagen tiempo-realización yendo de la mano izquierda proyectando a la derecha y ahí se nota las variaciones sinusoidales de cual hablamos anteriormente.

En la segunda se muestra la proyección de la imagen tiempo-realización de frente la cual seria la señal como se ve en la figura, pero teniendo en cuenta las variaciones dichas anteriormente.

Y en la tercera se muestra la proyección de la imagen tiempo-realización en el plano XY que seria proyectando de arriba hacia abajo también con las variaciones dichas anteriormente.

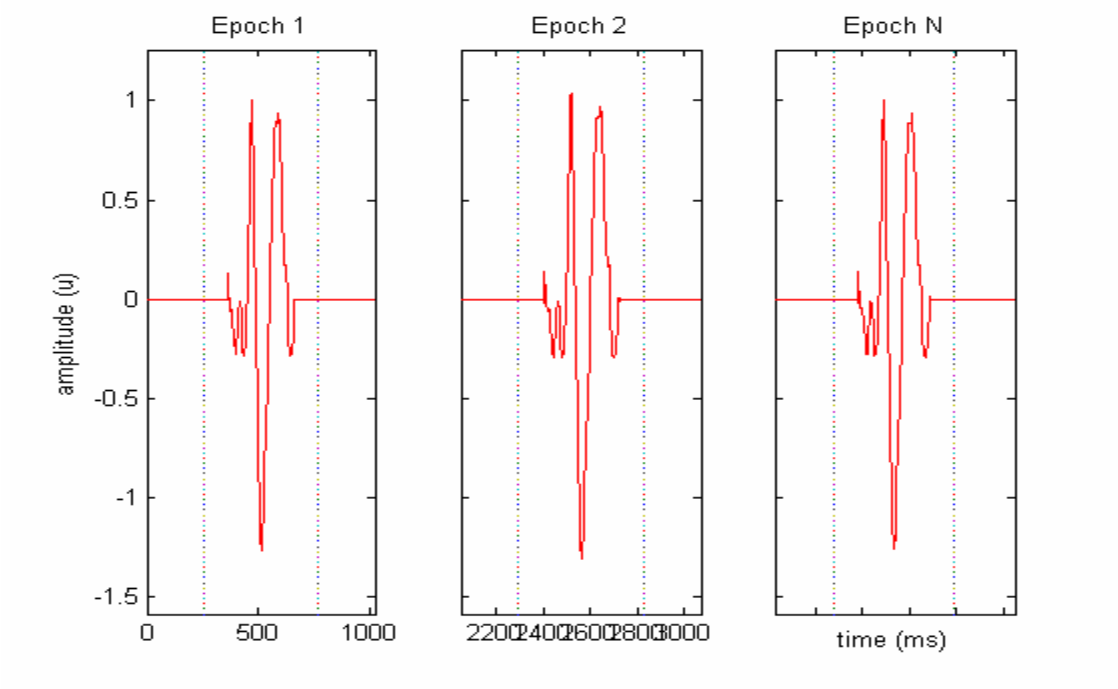

Figura 31. Las respuestas neuroeléctricas mostradas.

Ahí en la Figura 31 lo que se muestra es las respuestas neuroeléctrica (en este caso N respuestas) puesta una atrás de la otra.

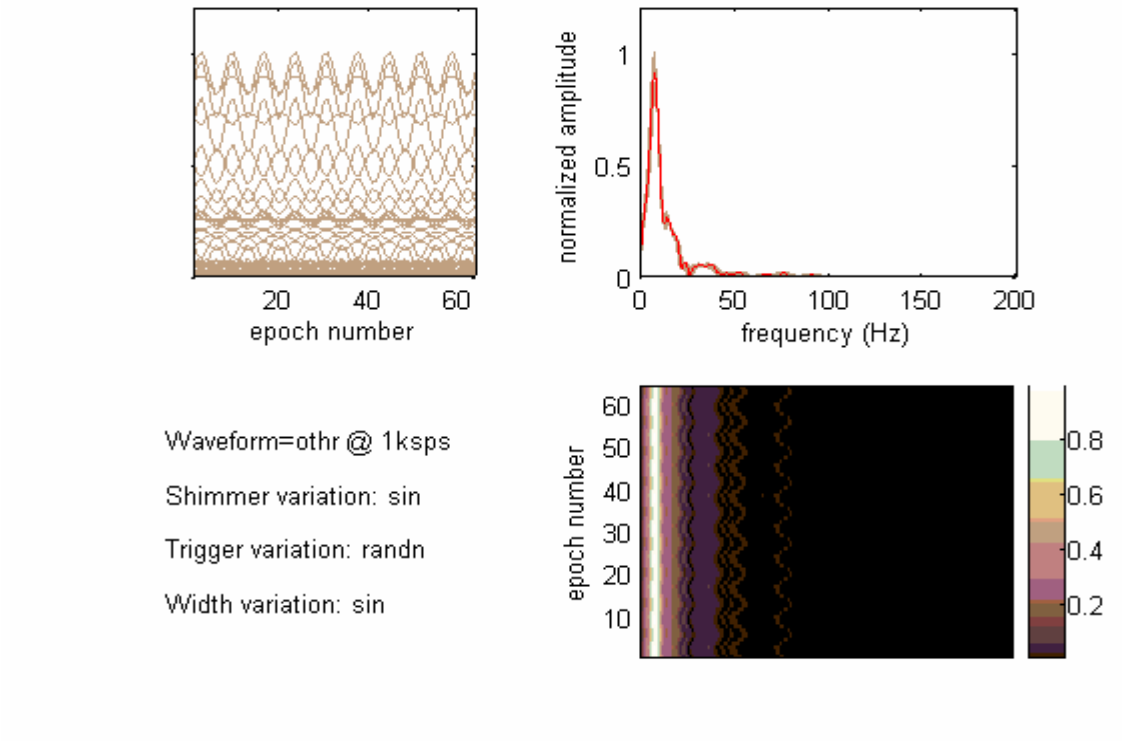

Figura 32. El espectro de frecuencia de la respuesta neuroeléctrica original.

En la Figura 32 vemos el espectro de frecuencia de la señal y nos damos cuenta que esta por debajo de 40 Hz aproximadamente y eso lo demuestra el artículo de donde sacamos la imagen del PEV. Cabe señalar que fijándonos en el espectro de frecuencia vemos que existen 2 colores que son el rojo y el gris, el rojo es debido a la respuesta neuroeléctrica original y el gris a como se afecta el espectro de frecuencia teniendo los parámetros *width modulation, shimmer y trigger.*

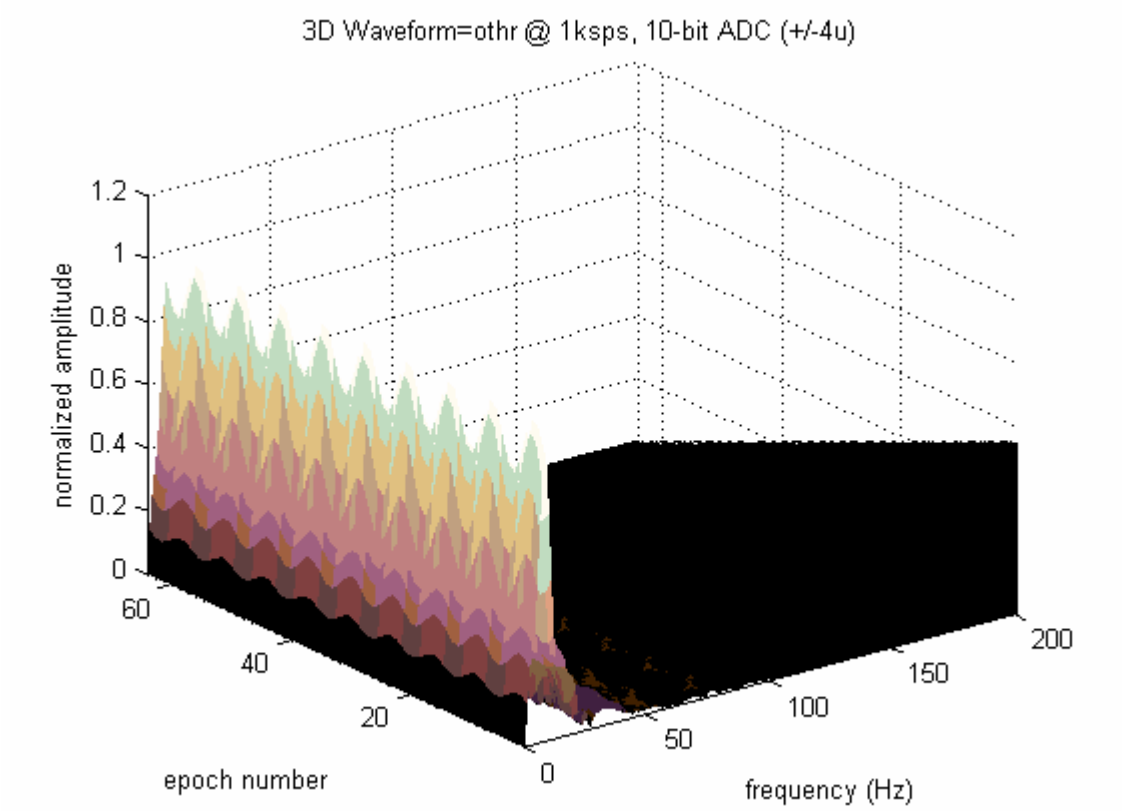

Figura 33. El espectro de frecuencia de todas las respuestas neuroeléctricas en una imagen 3D (en este caso son 64 respuestas).

En la Figura 33 se ve el espectro de frecuencia de todas las respuestas neuroeléctricas puesto en una imagen de 3D con las coordenadas de frecuencia, el número de la respuesta y finalmente la amplitud.

Recordamos que en todas las figuras mostradas, el número por defecto de la cantidad de bits del convertidor analógico digital el cual fue 10.

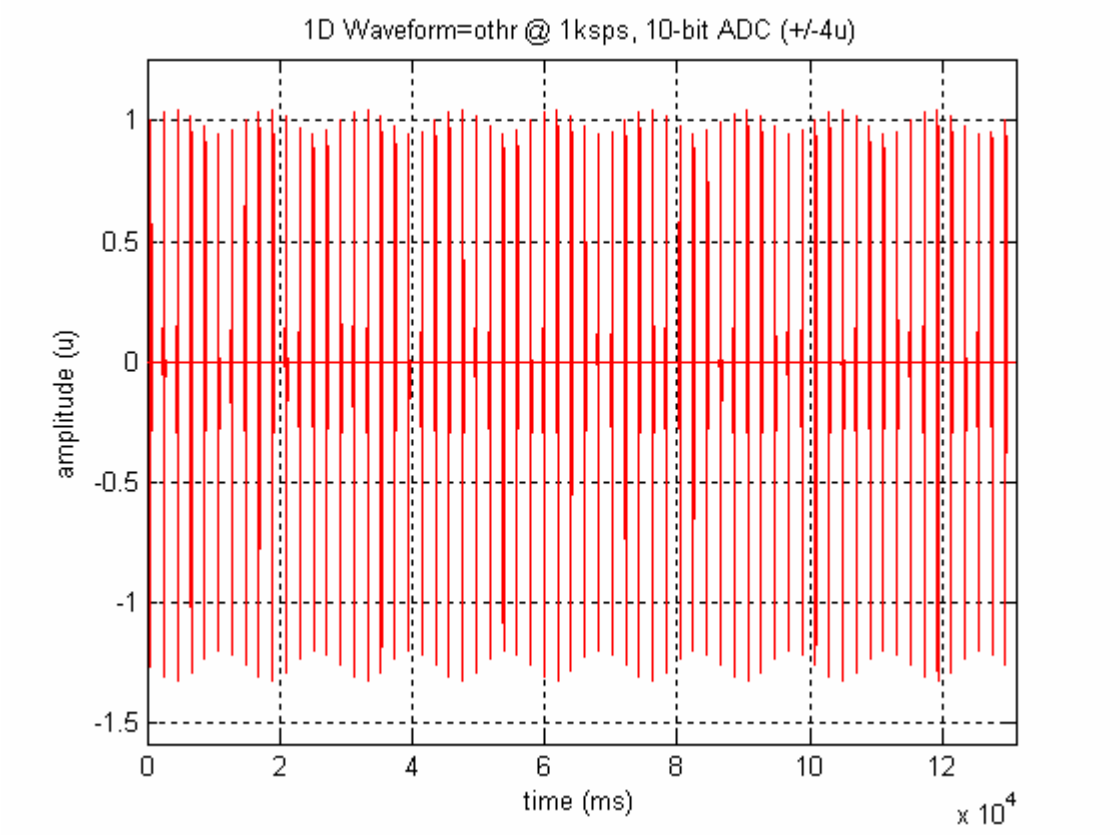

Figura 34. Las señales en tiempo real.

En la Figura 34 se muestra las señales en tiempo real o sea cuando se hace la prueba, lo que nos da realmente es esto.

En la Figura 35 se muestra la misma figura pero ampliada para que se pueda ver bien como es en la vida real.

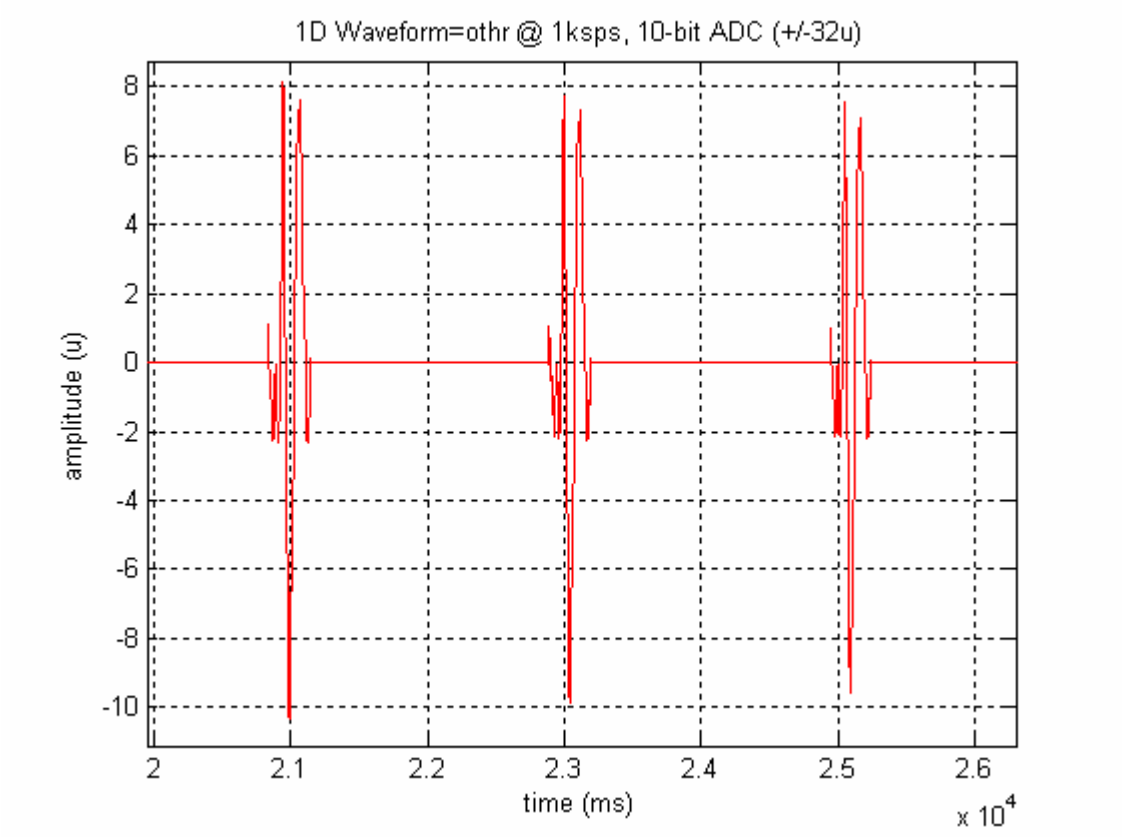

Figura 35. Señal que se obtiene en la vida real.

Nos damos cuenta también que los 3 parámetros que hablamos siguen afectando aquí las respuestas neuroeléctricas y si nos fijamos bien en la figura por arriba vemos como las modulaciones de amplitud es de forma sinusoidal y como dijimos que ese tipo de modulación se debe a la concentración del paciente.

Luego se escogió la misma señal y se le aplico el AVETESTNOISE, esta función hace lo mismo que AVETESTSIGNAL pero la diferencia esta en que con el AVETESTNOISE se le adiciona a la señal un ruido (el requerido por usted) y en este caso le adicionamos 2 interferencias, una a 60Hz con una amplitud de 0,4 y fase cero y otra de amplitud 0,4 y frecuencia de 180 Hz y como sabemos ese tipo de ruido siempre estas interferencias de la línea de alimentación afectan mucho a ese tipo de señales y se obtuvieron los resultados siguientes.

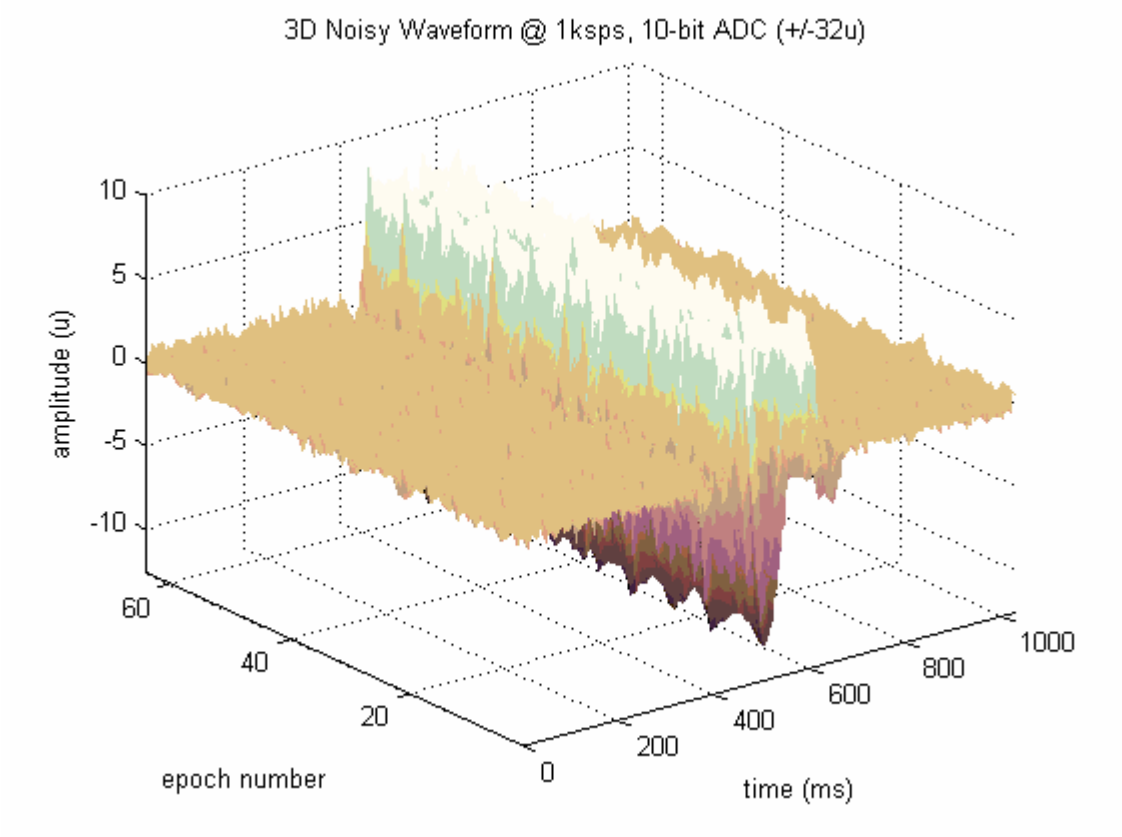

Figura 36. La imagen tiempo-realización con el ruido adicionado.

En la Figura 36 lo que se hizo fue coger las respuestas neuroeléctrica con el ruido adicionado y la interferencia también y ponerlas en una imagen 3D a la cual se le llama imagen tiempo-realización teniendo en cuenta las variaciones de *jitter*, *shimmer* (modulación de amplitud) y *width modulation* que fueron para esta prueba en cuanto al *shimmer* fue una variación sinusoidal y para el *width modulation* fue también sinusoidal y para el *trigger* fue aleatorio, también nos damos cuenta que las coordenadas de la imagen son el tiempo, el número de la respuesta (por defecto se tomo el valor 64) y la amplitud.

Señalamos que al igual que el AVETESTSIGNAL el AVETESTNOISE toma en cuenta la cantidad de bits del convertidor analógico digital que determina la resolución, como vemos que se tomo el valor por defecto que es 10 en este caso pero cabe señalar que para algunas aplicaciones especificas se exige el valor del convertidor analógico digital que normalmente es alrededor de 10.

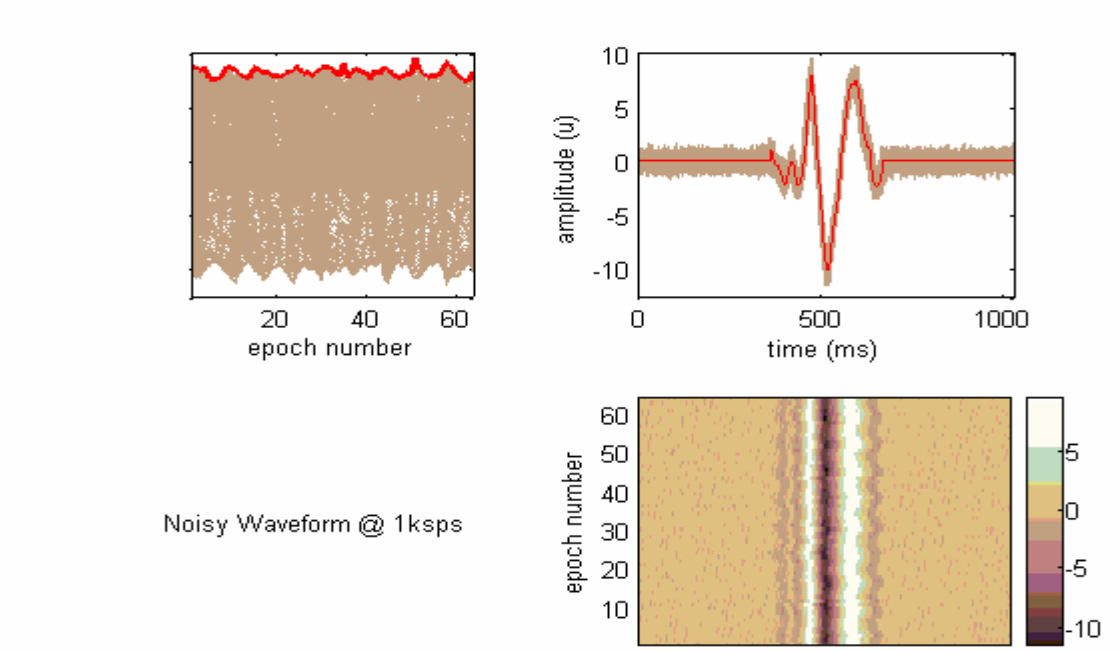

Figura 37. Las proyecciones de la imagen tiempo-realización en los 3 planos con el ruido adicionado.

En la Figura 37 se muestra las proyecciones de la imagen tiempo-realización en los 3 planos, XY, XZ, y YZ. En la primera se muestra la proyección de la imagen tiempo-realización yendo de la mano izquierda proyectando a la derecha y ahí se nota que las variaciones deberían ser sinusoidales pero teniendo el ruido adicionado también nos damos cuenta de la deformación de dichas variaciones. En la segunda se muestra la proyección de la imagen tiempo-realización de frente la cual seria la señal como se ve en la figura teniendo en cuenta las variaciones dichas anteriormente. En la tercera se muestra la proyección de la imagen tiempo-realización en el plano XY que seria proyectando de arriba hacia abajo también con las variaciones dichas anteriormente.

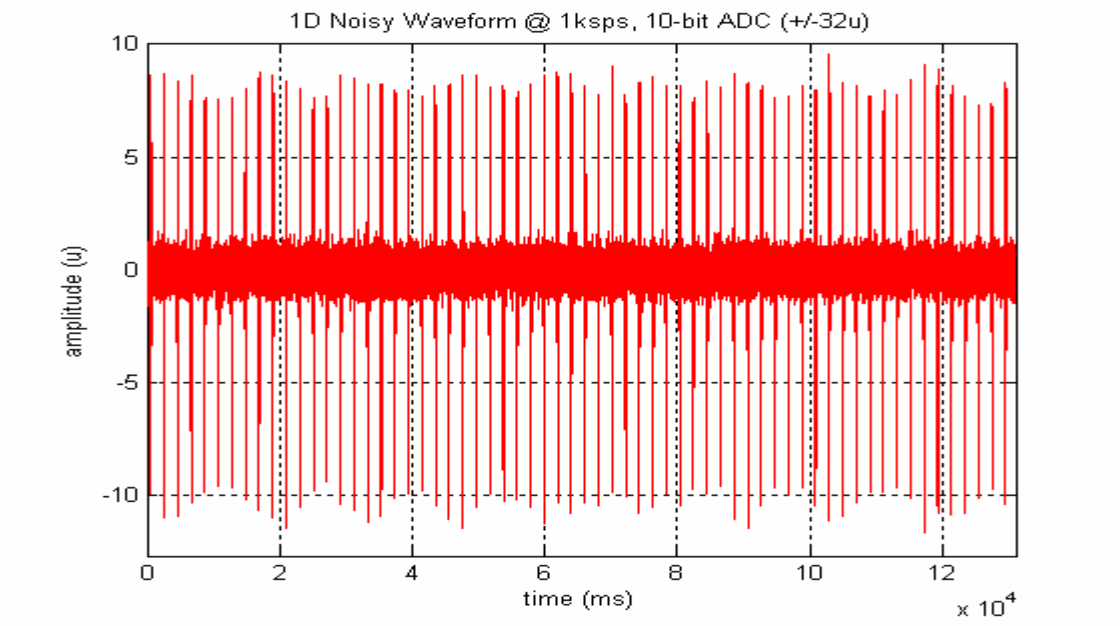

Figura 38. Las señales en tiempo real con el ruido adicionado.

Ahí es la señal verdadera (Figura 38) o sea lo que nos da en la vida real y ahí nos damos cuenta del nivel de ruido adicionado, Nos damos cuenta también que los 3 parámetros que hablamos al principio siguen afectando aquí las respuestas neuroeléctricas (estos parámetros afectan como un ruido mas aparte del que se le adiciono a la señal) y fijándonos bien vemos como las modulaciones de amplitud es de forma sinusoidal.

En la Figura 39 se muestra la misma figura pero ampliada para que se pueda ver bien como es en la vida real.

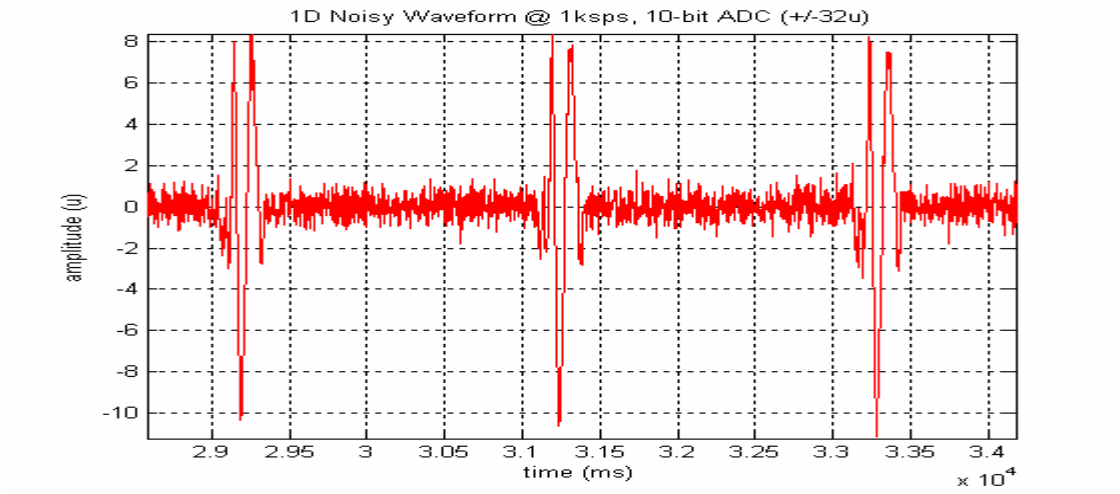

Figura 39. Señal simulada que se obtiene en la vida real.

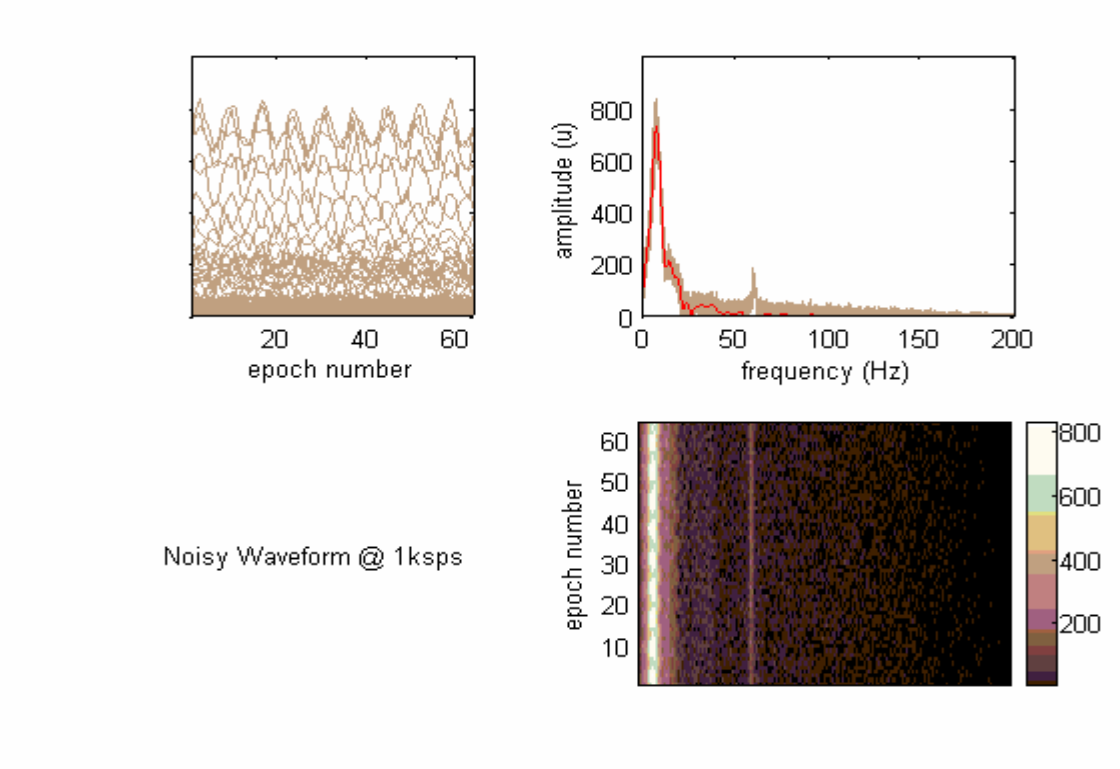

Figura 40. El espectro de frecuencia de la señal mezclada con el ruido.

Ahí en la Figura 40 se muestra el espectro de frecuencia de una sola respuesta con el ruido adicionado y ahí nos damos cuenta como el ruido afecta la señal en su espectro de frecuencia.

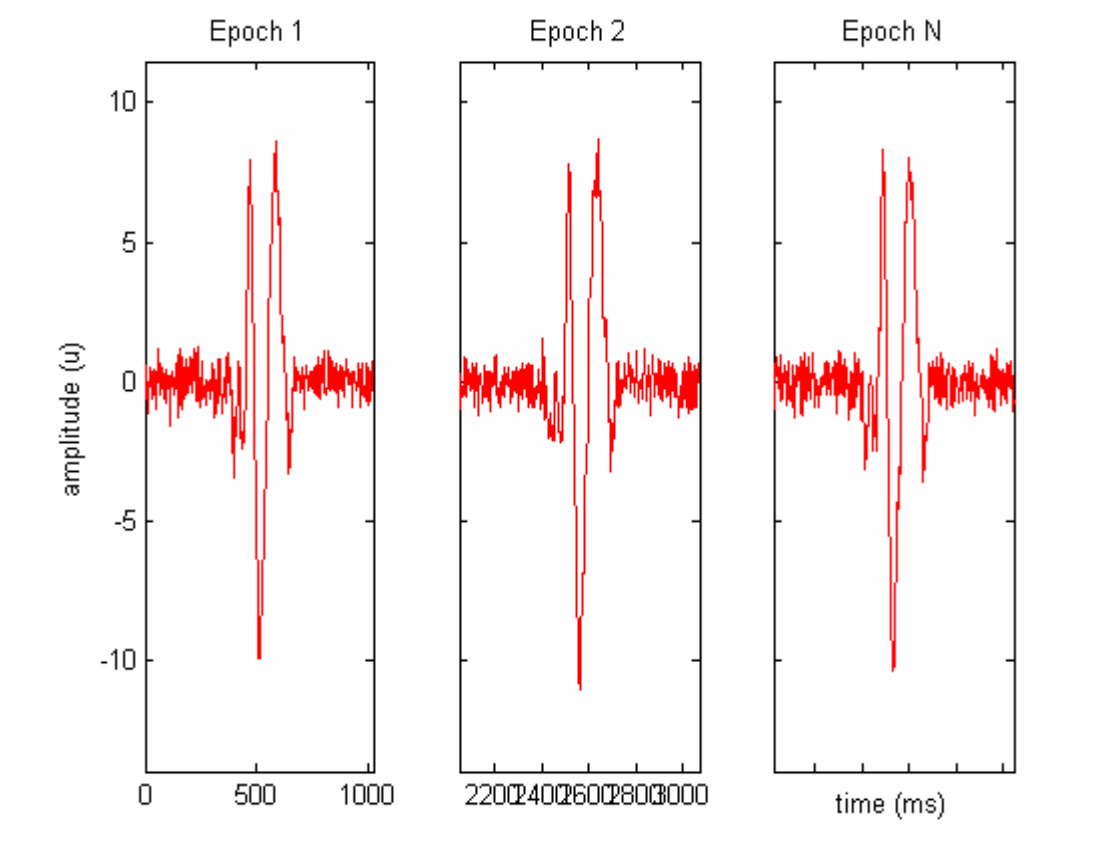

Figura 41. Las respuestas neuroeléctricas junto con el ruido mostradas una atrás la otra.

En la Figura 41 se ven las respuestas neuroeléctricas en tiempo real una atrás de la otra con el ruido adicionado teniendo también las afectaciones de la señal por los 3 parámetros dichos anteriormente.

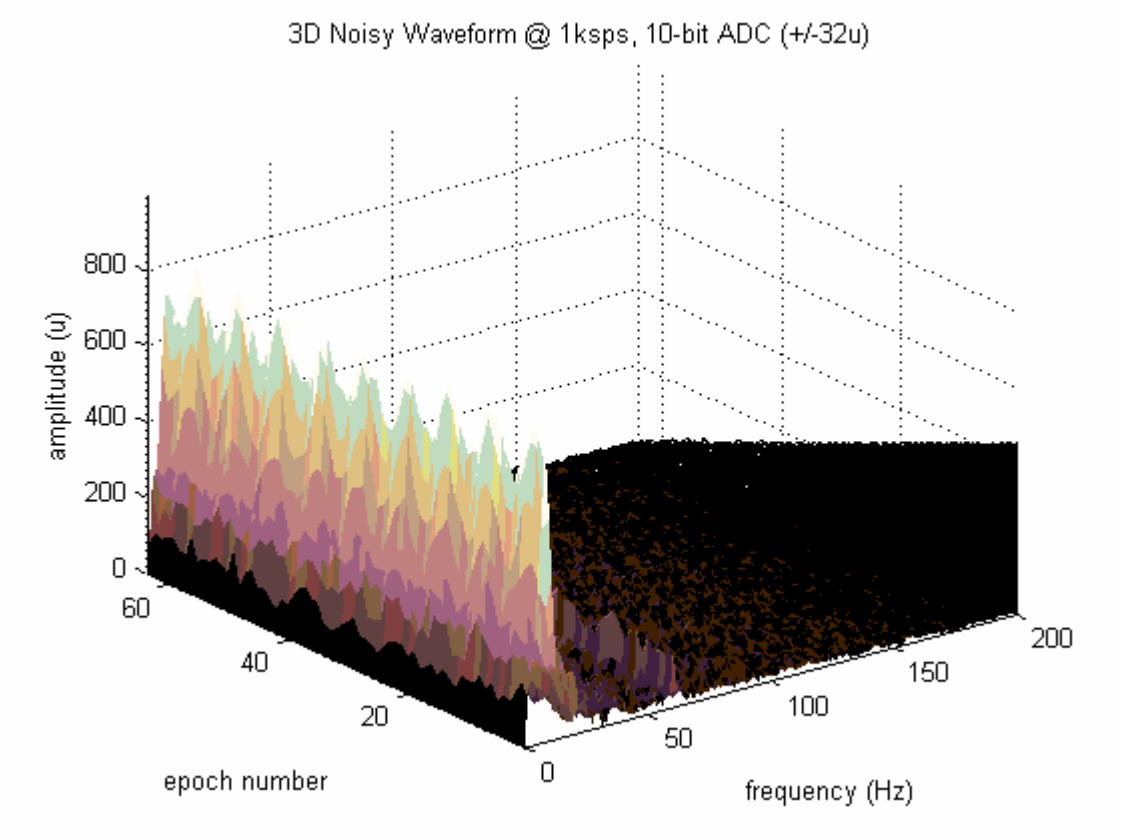

Figura 42. El espectro de frecuencia de todas las respuestas neuroeléctricas en una imagen 3D con el ruido adicionado.

En la Figura 42 se ve el espectro de frecuencia de todas las respuestas neuroeléctrica con el ruido adicionado puesto en una imagen de 3D con las coordenadas de, frecuencia, el número de la respuesta (por defecto 64) y finalmente la amplitud.

3.2 Cálculo de los coeficientes estadísticos de calidad de señal

Como ya sabemos, son estadígrafos aplicados para la evaluación cuantitativa de la calidad de respuesta evocada o para detectar la presencia del PE. Existen 3 indicadores que son CCR (coeficiente de correlación), SDR (cociente de desviación estándar), NRR (nivel de ruido residual).

Lo que se hizo fue tomar la señal obtenida en el capitulo 2 por el programa graph2vector y se le adicionó a la señal mediante el programa avetestnoise una interferencia a 60Hz que es común y que siempre se ve ese tipo de señal afectada por ese tipo de ruido y también se le adicionó una interferencia a 180Hz que es el tercer armónico de los 60Hz y sabemos que los armónicos impares afectan más la señal que los pares y se le calculo el coeficiente de

correlación pero se hizo el análisis para distintos tipos de promediado, los tipos de promediados que se usaron son, ensemble mean, trimmed mean, el modified trimmed mean, y el median filter.

Ensemble mean

El primer tipo de promediado que se usó es el ensemble mean y lo que hace es tomar los valores de cada columna de la matriz y calcular el promedio (sumar todos los valores y dividirlos entre la cantidad de muestras).

Trimmed mean

El segundo que se usó fue el trimmed mean y lo que hace es ordenar cada columna de la matriz en orden de menor a mayor y quitar el 5% de las muestras de arriba y de abajo y luego calcularle el promedio.

Modified trimmed mean

El tercero que se usó fue el modified trimmed mean y este lo que hace es mover la matriz unas cuantas muestras a la derecha y a la izquierda y la cantidad de muestras que hace falta mover se calcula mediante un análisis estadístico, y luego hace el promedio.

Median filter

El cuarto y el ultimo tipo de promediado que se usó fue es el median filter y lo que hace es tomar cada columna de la matriz y ordenarla y tomar el valor que esta en el medio.

Señalamos de nuevo que a la señal que se le hizo el análisis fue el VEP tipo flash.

Para calcular el coeficiente de correlación (CCR) y el cociente de desviación estándar (SDR) mediante el MATLAB7.0 se hicieron 2 programa que se llamaron ccrcoef y sdrcoef y se obtuvieron los resultados siguientes mostrados en tabla 1.

Tabla1. Los valores de CCR, Pn (µV2) Y SDR mediante 4 tipos de promediado.

	Ensemble mean	Trimmed mean	Modified trimmed mean	Median filter
CCR	0,9989	0,9988	0,9991	0,9980
P_n (μV^2)	0,0358	0,0335	0,0465	0,0345
SDR	12,6207	13,0416	11,0651	12,8510

Como sabemos que mientras el coeficiente de correlación es mas cercano a uno, el método de promediado es mejor y según tenemos visto en la tabla de los valores obtenidos nos damos cuenta que el modified trimmed mean es el mejor.

Pero en cuanto a la reducción del nivel de ruido, nos damos cuenta que en este caso el que se comportó mejor fue el trimmed mean, por eso hay que llegar a un compromiso o sea dependiendo de la prioridad que usted quiera, si usted da prioridad a la reproducibilidad pues se usa el modified trimmed mean ó al contrario se usa el trimmed mean. También se calculó la potencia del ruido en cada caso y es fácil notar que mientras más grande la potencia del ruido, el SDR correspondiente es menor.

Sabiendo que para el SDR la frontera aceptable es igual o mayor que 2 pues los valores que nos dieron son aceptable y la diferencia entre los cuatro tipo de promediado en cuanto al SDR no es grande pues recomendamos en general usar el modified trimmed mean.

También se calculó el NRR a través de un programa que se hizo y se llamó nrrcoef para los 4 tipos de promediados y nos dieron para los 4 casos el valor 3,4063 y nos damos cuenta primero de que cambiando el tipo de promediado no afecta este valor y también el valor indica que no es aceptable porque normalmente la frontera es entre 0 y 0,5 y esto se debe a que el ruido adicionado era muy grande que afectó mucho la señal y que hizo que este valor aumente mucho por encima lo normal.

Luego como vimos en el capítulo 2 cuando demostramos que con un filtro Butterworth de segundo orden y una frecuencia de corte de 20Hz pudimos disminuir mucho el nivel de ruido, a esa señal que es la misma con el cual siempre hicimos los análisis se le calculó el

NRR después de pasarla por el filtro y nos dio 1,8341 y ahí nos damos cuanta de lo mucho que bajó el NRR.

3.2.1 Afectaciones de width modulation, shimmer y trigger al PE

Como sabemos que estos parámetros afectan a la hora del promediado y pueden ser equivalente a un ruido adicionado, y para evaluarlo se necesita calcular el tercer indicador estadístico que es el NRR (nivel de ruido residual) y ahí hicimos algunas simulaciones de estos parámetros o sea teniendo una señal promediada se le adicionó modulación de ancho de forma sinusoidal, *shimmer* de forma sinusoidal, y *trigger* de forma aleatoria y sabemos que para calcular este indicador lo que se hace es sumar con signos alternos los registros individuales sucesivos de un PE. Por tanto la señal que simulamos no le adicionamos ruido solamente esta bajo los 3 parámetros indicados arriba y sabemos si cualquier señal no tiene ruido y no existe ningún efecto de lo que parámetros mencionados pues a la hora de alternar los signos el resultado teóricamente es cero.

Se calcularon los NRR para cada tipo de promediado mencionado antes y en los 4 casos nos dieron el mismo valor el cual fue 1,2616 y este según la frontera aceptable para este parámetro pues no es aceptable y se debe a que en las simulaciones de la señal, las afectaciones producidas por los 3 parámetros fueron grande que hicieron que este valor sea inaceptable. Luego se muestra las afectaciones producidas por estos parámetros a la señal en forma de ruido en la Figura 43.

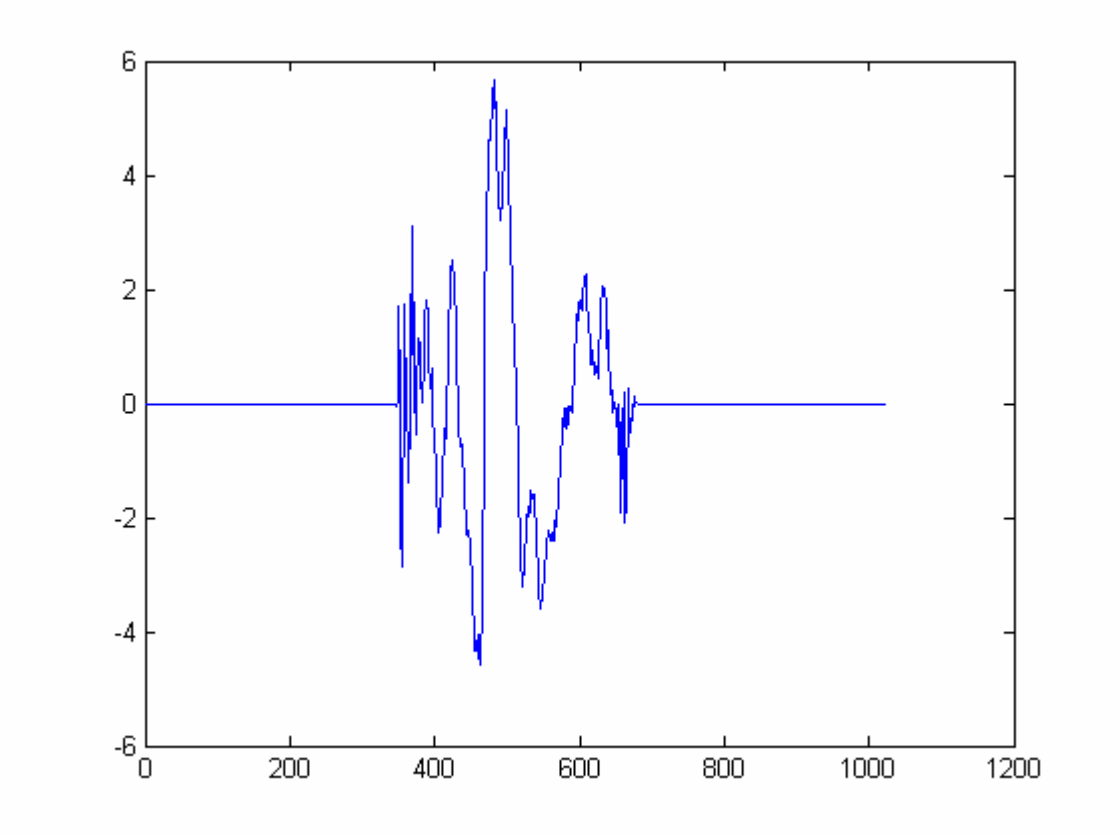

Figura 43. La equivalencia de las afectaciones de los parámetros como ruido.

Conclusiones

1- La metodología implementada y empleada en este trabajo permite obtener señales discretas (vector de valores) a partir de la representación gráfica (amplitud vs. tiempo) de las mismas.

2- La implementación de un banco de filtros permite mejorar la relación señal a ruido y llegar a un buen resultado.

3- La generación de imágenes tiempo-realización con características similares a las señales del mundo real ayuda a evaluar algoritmos relacionados con las ERSWS, específicamente a los relacionados con los PE.

4- Se programaron funciones para hacer cálculos de los 3 indicadores CCR, SDR, NRR que son de gran importancia en las señales de PE para explicar los resultados obtenidos con el promediado.

5- En las evaluaciones realizadas parece ser que el promediado recortado modificado supera a los demás métodos de promediado programados (promediado simple, promediado recortado y mediana) en cuanto a reproducibilidad, pero no así en cuanto a reducción del nivel de ruido, donde se comportó mejor el trimmed mean.

6- Las variaciones de ancho (width modulation) y amplitud (shimmer) de las ondas, así como el desplazamiento relativo de las mismas (trigger) en la señal de PE producen un afecto adicional de ‘ruido’.

Recomendaciones

Con vistas a trabajos de investigación futuros en esta temática, se recomienda:

- Que se continué el estudio e implementación de otros algoritmos de detección de potenciales evocados de difíciles acceso.
- Que, después de lograr tener una base de datos de imágenes tiempo-realización, aplicarle técnicas de procesamiento de imágenes y ver si con pocas repeticiones se logra un resultado satisfactorio.

Referencias

[1] J.A. Biurrun Manresa, “Filtrado digital de potenciales evocados auditivos de tronco cerebral”, disponible en

www.cori.unicamp.br/jornadas/completos/UNER/BIURRUN%20MANRESA,%JOSE.DOC , consultado en mayo 2007.

[2] J.J. Barajas de Prat, “Potenciales evocados auditivos continuos”, *revista electrónica de audiología*, Vol. 1,11 de marzo de 2002.

[3] N.A. Busch, S. Debener, C. Kranczioch, A.K. Engel, C.S. Herrmann, “Size matters: effects of stimulus size, duration and eccentricity on the visual gamma-band response”, *Clinical neurophysiology*, Vol. 115, pp. 1810-1820, 2004.

[4] J.M. Cornejo, M. Palacios, P. Granados, N. Castañeda, M. Cadena, “Potenciales evocados auditivos, respuesta de baja frecuencia para tonos de ráfagas de larga duración”, 2001, disponible en: www.hab2001.sld.cu/arrepdf/00186.pdf, consultado en mayo 2007.

[5] L.A.Cerquera Escobar, M.A.Para Rodríguez, “Unidades Motoras y Potenciales Evocados”, disponible en: www.paginas.usco.edu.co/~unibiomedica/linea3.html, consultado en mayo 2007.

[6] F.Z. Castro, R.F. Belda, J.J. Barajas de Prat, “La adaptación audioprotésica pediátrica precoz a partir de registros de potenciales evocados auditivos de estado estable”, *Acta otorrinolaringol*, Vol. 57, pp. 388–393, 2005.

[7] J.D. Hernández, F.Z. Castro, J.J. Barajas de Prat, “Normalización de los potenciales evocados auditivos del tronco cerebral 1, resultados en una muestra de adultos normoyentes” *Clínica Barajas España*, disponible en: www.auditio.com/revista/vol2/1/020104.pdf, consultado en mayo 2007.

[8] W. Jakuczun, E. Kublik, D.K. Wojcik, A. Wrobel, “Local classifiers for evoked potentials recorded from behaving rats”, *Acta neurobiol*, Vol. 65, pp. 425-434, 2005.

[9] M. Korostenskaja, K. Dapsys, A. Siurkute, V. Maciulis, O. Ruksenas, S. Kahkonen, “Evaluation of the new parameters of auditory evoked potentials (AEPs) in patients with schizophrenia spectrum disorders”, *ISNN 1392-0359*, Vol. 33, 2006.

[10] J.V. Lorenzo Ginori, A. Taboada Crispi, D.F. Lovely, “Ventricular late potential detection via image processing techniques”.

[11] K.J. Mobley, E. Gibson, “Potenciales evocados con tono BURST, aplicaciones clinicas”, disponible en*:* www.audicenter.com.ar/potenciales%20evocados.pdf, consultado en mayo 2007.

[12] T. Meigen, M. Bach, “Perceptual ranking versus visual evoked potencials for different local features in texture segregation”, disponible en*:* www.blackwell-synergy.com/doi/pdf/10.1111/j.1600-0420.2006.00812.x, consultado en mayo 2007.

[13] F. Miyara, “Seminario Taller Sobre Potenciales Evocados Auditivos”, disponible en: www.eie.fceia.unr.edu.ar/~acustica/biblio/evocado.pdf, consultado en mayo 2007.

[14] P. Martinez Beneito, A. Morant. Ventura, M.I. Pitarch Ribas, F.J. Garcia Callejo, J. Marco Algarra, "Potenciales evocados auditivos de estado estable a multifrecuencia como tecnica de determinación de umbrales auditivos", *Acta otorrinolaringol*, Vol. 53, pp. 707–717, 2002.

[15] E.A. Milan, Y. Morales, L. Vaillant, R. Sanchez, A. Montoya, "Modelo en diferencias finitas para la actividad eléctrica cerebral evocada por estímulos periódicos", 2003, disponible en: www.memsocbio.sld.cu/habana2003/Articles/T_0062.pdf, consultado en mayo 2007.

[16] M.C. Niño de Mejia, "Potenciales evocados somatosensoriales", disponible en

www.scare.org.co/comites_ocg/recursos/potenciales.pdf, consultado en mayo 2007.

[17] J.V. Odom, M. Bach, C. Barber, M. Brigell, M.F. Marmor, A. Patrizia Tormene, G.E. Holder & Vaegan, "Visual evoked potentials standard (2004)", *Documenta Ophthalmologica*, Vol. 108, pp. 115–123, 2004.

[18] E.Ojeda, "Aspectos metodológicos", disponible en

www.acnweb.org/pub/guia/g7cap14.pdf, consultado en mayo 2007.

[19] M.A. Parra, "Método para el estudio de los movimientos oculares", *Colombia médica*, Vol. 35, No. 2, 2004.

[20] A.M. Rodríguez, "Potenciales Evocados Auditivos", disponible en www.phonak-pip.com/pdfss/alberto%20mateos%20potenciales%20evocados.pdf, consultado en mayo 2007.

[21] S. Sayas, "Procesamientos posteriores a la etapa de adquisición de señales nerviosas en dispositivos implantables neuronales", disponible en www.nib.fmed.edu.uy/Seminario2005/monografias2005/sayas.pdf, consultado en mayo 2007.

[22] D.C. Sanchez Sainz-trapaga, "Potenciales evocados auditivos de tronco cerebral en recien nacidos", *Tesis realizada en el servicio de neonatolgia hospital clínico universitario san carlos madrid*, 1994

[23] M.C. Tapia Toca, G. Savio López, "Potenciales evocados auditivos de estado estable en el estudio de dos pacientes con neuropatia auditiva", *Acta otorrinolaringol*, Vol. 56, pp. 240–245, 2005.

[24] E. Vaquero, J. Cardoso, M. Vázquez, "Potenciales evocados visuales por un proceso de atención y diferencias de genero", disponible en

www.psicologia-online.com/ciopa2001/actividades/13/index.html, consultado en mayo 2007.

[25] E. Vannier, O. Adam, J. Francois Motsch, "Objective detection of brainstem auditory evoked potencials with a priori information from higher presentation levels", *Artificial intelligence in medicine*, Vol. 25, pp. 283–301, 2002.

[26] C.E. Vasios, G.K. Matsopoulos, K.S. Nikita, N. Uzunoglu, "Classification of event-related potentials using multivariate autoregressive modelling combined with simulated annealing ", *Journal of automatic control*, Vol. 13,pp. 7-11, 2006.

[27] C. Vasios, C. Papageorgiou, G.K. Matsopoulos, K.S. Nikita, N.Uzunoglu, “A decision support system of evoked potentials for the classification of patients with first –episode schizophrenia”, disponible en: www.gipsy.uni.geottingen.de, consultado en mayo 2007.

[28] I. Zamarbide, “Potenciales evocados somatosensitivos”, disponible en www.elmundo.es, consultado en mayo 2007.

[29] “Potenciales Evocados”, disponible en

www.san.gva.es/comun/ciud/docs/pdf/neurofisiologia6c.pdf, consultado en mayo 2007.

[30] “Potenciales Evocados Auditivos de Estado Estable”, disponible en

www.acta.otorrinolaringol.esp.medynet.com/textocompleto/actatorrino45/3.pdf, consultado en mayo 2007.

[31] “Voces en el silencio: potenciales evocados auditivos de estado estable”, disponible en www.vocesenelsilencio.org.ar/modules.php?name=News&file=article&sid=193-35k-3jun, consultado el 3 de junio del 2007.

[32] “Potenciales evocados”, disponible en

www.bibliodgsca.unam.mx/tesis/tes8fecv/sec_2.htm-2k, consultado en mayo 2007.

[33] “Potenciales evocados auditivos de tronco cerebral en la hipoacusia”, disponible en: www.scielo.cl/pdf/rcp/v61n6/art05.pdf, consultado en mayo 2007.

[34] “Potenciales evocados visuales”, disponible en http://enciclopedia.us.es/index.php/Potenciales_evocados_visuales, consultado en mayo 2007.

Anexos

Aquí se muestran los programas que se usaron en la tesis. En cada programa, la ayuda está marcada en verde. Se empleó el MATLAB7.0 por ser una herramienta potente de cálculo y de simulación.

Programa avetestsignal:

```
function [svector,smatrix,refm,rsa,lsa,amplits,widths,delays,fs,nb,Amax] =
avetestsignal(wave,wp,A,fs,per,N,aml,pam,wml,pwm,pml,ppm,pp,nb,Amax,os);
%AVETESTSIGNAL generates event-related signals in a wide sense (ERSWS)
%  to study signal 'averaging' methods of detection/analysis.
%  Signals are generated from a basic waveform, given the shimmer, jitter,
%  and width modulation, affecting the basic waveform in every event (epoch)
%
%[svector,smatrix,refm,rsa,lsa,amplits,widths,delays,fs,nb,Amax] =
%avetestsignal(wave,wp,A,fs,per,N,aml,pam,wml,pwm,pml,ppm,pp,nb,Amax,os);
%
%  Inputs:
%    wave = waveform
%                'db1' = first order Daubechies wavelet or Haar wavelet
%                'db2' = second order Daubechies wavelet ...
%                'dbO' = Daubechies wavelet of order O (O is integer, 1<O<10)
%                'sym2' = second order Symlets wavelet
%                'sym3' = third order Symlets wavelet ...
%                'symO' = Symlets wavelet of order O (O is integer, 2<O<8)
%                'coif1' = first order Coiflets wavelet
%                'coif2' = second order Coiflets wavelet ...
%                'coifO' = Coiflets wavelet of order O (O is integer, 1<O<5)
%                'gaus1' = first order Gaussian wavelet (default value)
%                'gaus2' = second order Gaussian wavelet or Mexican hat ...
%                'gausO' = Gaussian wavelet of order O (O-th derivative)
%                'meyr' = Meyer wavelet
%                'morl' = Morlet wavelet
%                'othr' = other waveform (a vector has to be passed to wp)
%    wp = waveform parameters
%             if wave is a wavelet, then wp = iter (the number of samples of
%                the wavelet will be forced to 2^iter), default wp = 8
%             if wave=='othr', then wp = vector with the waveform, (the
%                number of samples should be a power of 2, or slightly lower,
%                otherwise a zero padding will be done to complete it
```

```
%                    truncated (if >)
%             if aml=='nul', then pam = 0
%    wml = width modulation law
%          'null' = no width modulation
%          'sin' = sinusoidal variation around the width of the initial wave
%                                                              (w0, in ms)
%          'rand' = random variation (uniform distribution) around w0
%          'randn' = random variation (normal distribution) around w0
%          'othr' = other variation law around w0 (a vector is passed to pwm)
%    pwm = width modulation parameter(s)
%             if wml=='sin', then pwm = [ra rf rp], where
%                ra = relative amplitude, fraction of w0 (0 < ra < 0.5)
%                rf = relative frequency (0 < rf < 0.5)
%                rp = relative phase (0 < rp < 2pi)
%             if wml=='rand', then pwm = rv, where rv is the relative
%                variation, as a fraction of w0 (0 < rv < 0.5)
%             if wml=='randn', then pwm = rsd, where rsd is the relative
%                standard deviation, as a fraction of w0 (0 < rsd < 0.1)
%             if wml=='othr', then pwm = vector with N values between
%                -w0/2 and w0/2, otherwise it is composed by
%                concatenation (if <), or truncated (if >)
%             if wml=='nul', then pwm = 0
%    pml = position modulation (jitter) law
%          'null' = no position modulation
%          'sin' = sinusoidal variation around the position of initial wave
%          'rand' = random variation (uniform distribution) around t0
%          'randn' = random variation (normal distribution) around t0 (def.)
%          'othr' = other variation law around t0 (vector is passed to ppm)
%    ppm = position modulation parameter(s)
%             if pml=='sin', then pwm = [ra rf rp], where
%                ja = jitter amplitude, in ms (default = 2ms)
%                rf = relative frequency (0 < rf < 0.5)
%                rp = relative phase (0 < rp < 2pi)
```

```
%            if pml=='rand', then ppm = rv, where rv is the relative
%                    variation, as a fraction of w0 (0 < rv < 0.5)
%            if pml=='randn', then ppm = rsd, where rsd is the relative
%                    standard deviation, as a fraction of w0 (0 < rsd < 1/10)
%            if pml=='othr', then ppm = vector with N values between
%                    -w0/2 and w0/2, otherwise it is composed by
%                    concatenation (if <), or truncated (if >)
%            if pml=='nul', then ppm = 0
%   pp = pivot position or position taken as a reference for the wave
%            pp = 'c' when the pivot is the center of the waveform (default)
%            pp = 'o' when the pivot is the onset of the waveform
%   nb = number of bits of the analog to digital converter (ADC) block,
%            which determines the resolution (default value is nb=10)
%   Amax = maximun value that can be converted by the ADC
%            (default value is Amax=4*A)
%   os = oversampling factor to get the 'continuos' waveforms
%            (signals are oversampled at 2^os times), default value is os=4
%
% Outputs:
%   svector = column vector of ERSWS, showing the 'isoelectric' intervals,
%                that is, those segments between the analysis windows.
%                Therefore, the total number of points in svector is
%                PER times higher than in smatrix
%   smatrix = N-row matrix of the ERSWS, windowed for analysis,
%                where the first row represents the reference waveform
%                centered in the analysis window of twice the width
%                of the basic waveform
%   refm = position of reference marks (fiducial marks for segmentation)
%   rsa = number of samples to the right of fid. marks, default=156
%   lsa = number of samples to the left of fid. marks, default=99
%   amplits = column vector of maximun amplitudes of the N ERSWS, in u
%   widths  = column vector of widths of the N ERSWS, in ms
%   delays  = column vector of delays respect to t0 of the N ERSWS, in ms
```

```
%    inputs fs, nb, Amax are given as outputs as well...
% Plots:
%    -3D representation of smatrix in the time domain
%    -Waveforms (XZ plane = amplitude vs. time), highlighting basic waveform
%    -Amplitudes and shimmer (YZ plane = amplitude vs. epoch number)
%    -Widths, positions and jitter (XY plane = epoch number vs. time)
%    -svector (amplitude vs. time)
%    -3D representation of smatrix in the frequency domain
%    -Waveforms in the frequency domain (XZ plane = amplitude vs. frequency)
%    -Amplitudes and shimmer in frequency domain (YZ plane)
%    -Widths, positions and jitter in the frequency domain (XY plane)



%--------------------------------------------------------------------------
% default values
if (nargin<1)|(isempty(wave)),
    wave='gaus1';    % default waveform (1st order Gaussian wavelet)
end
if (nargin<2)|(isempty(wp)),
    wp=7;            % default number of iterations (waveform of 128 points)
end
if (nargin<3)|(isempty(A)),
    A=1000;           % default amplitude value (A = 1000 u)
end
if (nargin<4)|(isempty(fs)),
    fs=1;            % default sampling frequency (fs = 1 ksps)
end
if (nargin<5)|(isempty(per)),
    per=2;           % default period (per = 2 times analysis window length)
end
if (nargin<6)|(isempty(N)),
    N=64;           % default number of events (N = 64)
```

```
end
if (nargin<7)|(isempty(aml)),
    aml='sin';       % default AM law (sinusoidal)
end
if (nargin<8)|(isempty(pam)),
    pam=0.05; % default AM parameter (modulation of 5%)
end
if (nargin<9)|(isempty(wml)),
    wml='sin';       % default width mod law (sinusoidal)
end
if (nargin<10)|(isempty(pwm)),
    pwm=0.05;        % default width mod parameter (modulation of 5%)
end
if (nargin<11)|(isempty(pml)),
    pml='randn';     % default delay modulation law (random Gaussian)
end
if (nargin<12)|(isempty(ppm)),
    ppm=2;           % default delay modulation parameter (std=2ms)
end
if (nargin<13)|(isempty(pp)),
    pp='c';           % default pivot position (center of the waveform)
end
if (nargin<14)|(isempty(nb)),
    nb=10;           % default number of bits of ADC (nb=10)
end
if (nargin<15)|(isempty(Amax)),
    Amax=4*A;        % maximun value that can be converted by the ADC
end
if (nargin<16)|(isempty(os)),
    os=4;            % oversampling to get 'continuos' signals (16x)
end
```

```
% if 2^fix(log2(N))~=N,
%     N=2^(fix(log2(N))+1);       % forces N to the next power of two
% end

if aml(1:3)=='sin',              % if AM law is sinusoidal
    if length(pam)<1,
        ra=0.05;                 % default relative amplitude = 0.05
        pam(1)=ra;
    end
    if length(pam)<2,
        rf=1/7;                  % default relative freq = 1/7
        pam(2)=rf;
    end
    if length(pam)<3,
        rp=0;                    % default relative phase = 0
        pam(3)=rp;
    end
    am=pam(1)*sin(2*pi*pam(2)*(0:N-1)+pam(3));
elseif aml(1:3)=='ran',
    if aml=='randn',
        am=pam*randn(1,N);
        am(am<-1)=-1;
        am(am>1)=1;
    else,
        am=2*pam*rand(1,N)-pam;
    end
elseif aml(1:3)=='nul',
    am=zeros(1,N);
elseif aml(1:3)=='oth',
    [r,c]=size(aml);
    if r>c, aml=aml'; end        % forces to row vector
    while length(aml)<N,
        aml=[aml aml];           % concatenates
```

```
    end

    am=pam(1:N);                    % truncates
end
am(am>A)=A;
am(am<-A)=-A;

if wml(1:3)=='sin',                 % if WM law is sinusoidal
    if length(pwm)<1,
        ra=0.05;                    % default relative amplitude = 0.05
        pwm(1)=ra;
    end
    if length(pwm)<2,
        rf=1/7;                     % default relative freq = 1/7
        pwm(2)=rf;
    end
    if length(pwm)<3,
        rp=0;                       % default relative phase = 0
        pwm(3)=rp;
    end
    wm=pwm(1)*sin(2*pi*pwm(2)*(0:N-1)+pwm(3));
elseif wml(1:3)=='ran',
    if wml=='randn',
        wm=pwm*randn(1,N);
        wm(wm<-0.5)=-0.5;
        wm(wm>0.5)=0.5;
    else,                           % wml='rand',
        wm=2*pwm*rand(1,N)-pwm;
    end
elseif wml(1:3)=='nul',
    wm=zeros(1,N);

elseif wml(1:3)=='oth',
    [r,c]=size(wml);
    if r>c, wml=wml'; end           % forces to row vector
    while length(wml)<N,
        wml=[wml wml];              % concatenates
    end
    wm=pwm(1:N);                    % truncates
end

if pml(1:3)=='sin',                 % if delay mod law is sinusoidal
    if length(ppm)<1,
        ja=2;                       % default relative amplitude = 2ms
        ppm(1)=ja;
    end
    if length(ppm)<2,
        rf=1/7;                     % default relative freq = 1/7
        ppm(2)=rf;
    end
    if length(ppm)<3,
        rp=0;                       % default relative phase = 0
        ppm(3)=rp;
    end
    pm=ppm(1)*sin(2*pi*ppm(2)*(0:N-1)+ppm(3));
elseif pml(1:3)=='ran',
    if pml=='randn',
        pm=ppm*randn(1,N);
%         pm(pm<-0.5)=-0.5;
%         pm(pm>0.5)=0.5;
    else,                           % pml='rand',
        pm=2*ppm*rand(1,N)-ppm;
    end
elseif pml(1:3)=='nul',
    pm=zeros(1,N);
```

```
elseif pml(1:3)=='oth',
     [r,c]=size(pml);
     if r>c, pml=pml'; end        % forces to row vector
     while length(pml)<N,
          pml=[pml pml];          % concatenates
     end
     pm=ppm(1:N);                 % truncates
end
%-------------------------------------------------------------------------
% Basic waveform and 'continuos' basic waveform
if wave(1:3)=='oth',             % if the waveform is given as a vector wp,
     x=wp;                       % basic waveform
     [r,c]=size(x); if r>c, x=x'; end
     long=length(x);             % length of x is forced to the
     lon=2^ceil(log2(long));     % next power of two by zero padding
     zer=lon-long;               % zeros to pad (half at both sides)
     x=[zeros(1,round(zer/2)) x zeros(1,zer-round(zer/2))]; % zero padding
     xc=interpolation(x,2^os);   % obtains the initial 'continuous' waveform
else,                            % if the basic waveform is a wavelet,
     x=mywavefun(wave,wp);       % obtains the basic waveform
     xc=mywavefun(wave,wp+os);   % obtains the initial 'continuous' waveform
end
xc=xc-mean(xc); x=x-mean(x);     % cancels DC level out
xc=A*xc/max(xc); x=A*x/max(x);   % forces amplitude to A
w0=length(x)/fs;                 % width of the basic waveform, in ms
wm(wm>w0/2)=w0/2;
wm(wm<-w0/2)=-w0/2;
pm(pm>w0/2)=w0/2;
pm(pm<-w0/2)=-w0/2;
%-------------------------------------------------------------------------
% Basic waveforms in analysis windows
xw=[zeros(1,length(x)/2) x zeros(1,length(x)/2)];
xcw=[zeros(1,length(xc)/2) xc zeros(1,length(xc)/2)];    % 'continuos'
```

```
%-------------------------------------------------------------------------

% Matrix
smatrix=zeros(N,length(xw));   % allocated matrix
smatrixc=zeros(N,length(xcw)); % allocated matrix
onsets=zeros(N,1);             % allocated matrix
onsets(1,1)=length(xw)/4;
if pp=='o',              % waveform onset as pivot
    pivotc=length(xcw)/4;   pivot=length(xw)/4;
else,                    % center of waveform as pivot
    pivotc=length(xcw)/2;   pivot=length(xw)/2;
end
for i=2:N,
    L=round(100*(1+wm(i))); M=100;
    tmp=resample(xc,L,M);    % shrinked/expanded basic 'continuos' wave
    if pp=='o',              % waveform onset as pivot
        ini=round(pivotc+(2^os)*pm(i)/fs)+1;
        pm(i)=(ini-pivotc-1)/2^os; % corrected delay according to delta-t min
    else,                    % center of waveform as pivot
        ini=round((pivotc-length(tmp)/2+(2^os)*pm(i))/fs)+1;
        pm(i)=(ini-pivotc+length(tmp)/2-1)/2^os; % corrected delay according to delta-t min
    end
    en=ini+length(tmp)-1;
    smatrixc(i,ini:en)=(1+am(i))*tmp;
%     smatrix(i,:)=decimation(smatrixc(i,:),2^os);
    smatrix(i,:)=decimate(smatrixc(i,:),2^os);
    onsets(i,1)=ini/2^os; % corrected onsets according to delta-t min
end
smatrix(1,:)=xw;                          % 1st epoch is the basic waveform
% ADC quantization
lsb=Amax/2^(nb-1);
smatrix(2:N,:)=lsb*round(smatrix(2:N,:)/lsb);
smatrix(smatrix>Amax)=Amax;               % ADC range limitation
```

```
%-------------------------------------------------------------------------
% Modulation laws (values vs. realization number)
amplits=max(smatrix')';         % maximun amplitudes of the N ERSWS, in u
widths=(wm'+1)*w0;              % widths of the N ERSWS, in ms
delays=pm'+pivot;                     % delays of the N ERSWS, in ms
%-------------------------------------------------------------------------
% PLOTS
t=0:1/fs:size(xw,2)-1/fs;      % time axis (ms) for smatrix and analysis w.
tc=0:1/(fs*2^os):size(xcw,2)-1/(fs*2^os);   % 'continuos' time axis (ms)
t1D=0:1/fs:per*N*length(t);     % time axis (ms) for svector

svector=zeros(1,length(t1D));  % allocation vector
for i=0:N-1,
    svector(1,i*length(t)*per+1:i*length(t)*per+length(xw))=smatrix(i+1,:);
end
%-------------------------------------------------------------------------
%   refm = position of reference marks (fiducial marks for segmentation)
refm=size(smatrix,2)/2:per*size(smatrix,2):length(svector);
%   rsa = number of samples to the right of fid. marks
rsa=size(smatrix,2)/2;
%   lsa = number of samples to the left of fid. marks
lsa=size(smatrix,2)/2-1;
% 3D plot of smatrix
figure;                         % Fig. 1
surf(t,1:N,smatrix)
shading flat
% colormap(gray)
colormap(pink)
xlabel('time (ms)');
ylabel('epoch number');
zlabel('amplitude (u)');
title(['3D Waveform=',wave,' @ ',num2str(fs),'ksps, ',num2str(nb),'-bit ADC (+/-',num2str(Amax),'u)']);
axis([0 max(t) 1 N 1.2*min(svector) 1.2*max(svector)]);
```

```
figure;                                 % Fig. 2
% Amplitudes and shimmer (YZ plane = amplitude vs. epoch number)
subplot(221)

% surf(t,1:N,smatrix)
% shading flat
% colormap(pink)
% view(90,0);
% ylabel('epoch number');
% axis([0 max(t) 1 N 1.2*min(svector) 1.2*max(svector)]);
% set(gca,'ZTickLabel',{[]})

[ro,co]=size(smatrix);
if ro>=co,
     mplot=smatrix';
else,
     mplot=smatrix;
end
plot(1:N,mplot','Color',[207/255 169/255 146/255]); hold on
plot(1:N,max(smatrix'),'r','LineWidth',2); hold off % shimmer
xlabel('epoch number');
set(gca,'YTickLabel',{[]})
axis([1 N 1.1*min(svector) 1.1*max(svector)]);

set(gca,'PlotBoxAspectRatio',[1 1 1]);

% Waveforms (XZ plane = amplitude vs. time)
subplot(222)
% plot(t,smatrix,'Color',[.5 .5 .5]); hold on
plot(t,mplot,'Color',[207/255 169/255 146/255]); hold on
plot(t,xw,'r','LineWidth',1); hold off  % basic
xlabel('time (ms)');
```

```
ylabel('amplitude (u)');
axis([0 max(t) 1.2*min(svector) 1.2*max(svector)]);
% Widths, positions and jitter (XY plane = epoch number vs. time)
subplot(224)
pcolor(t,1:N,smatrix);
shading flat
% colormap(gray)
colormap(pink)
hold on
plot(onsets,1:N,'r','LineWidth',1);                    % onsetss
plot(onsets+widths,1:N,'r','LineWidth',1); hold off   % onsets+widths
ylabel('epoch number');
set(gca,'XTickLabel',{[]})
axis([0 max(t) 1 N]);
colorbar;
% Info
subplot(223)
message1=['Waveform=',wave,' @ ',num2str(fs),'ksps'];
message2=['Shimmer variation: ',aml];
message3=['Trigger variation: ',pml];
message4=['Width variation: ',wml];
text(0.05,0.8,message1);
text(0.05,0.6,message2);
text(0.05,0.4,message3);
text(0.05,0.2,message4);
set(gca,'XTickLabel',{[]})
set(gca,'YTickLabel',{[]})
axis off

% Plot of svector (amplitude vs. time)
figure;                                % Fig. 3
plot(t1D,svector,'r'); grid
xlabel('time (ms)');
```

```
ylabel('amplitude (u)');
title(['1D Waveform=',wave,' @ ',num2str(fs),'ksps, ',num2str(nb),'-bit ADC (+/-',num2str(Amax),'u)']);
axis([0 max(t1D) 1.2*min(svector) 1.2*max(svector)]);

% Plot of svector (amplitude vs. time) for epochs 1, 2, and N
figure;                        % Fig. 4
subplot(131)
plot(t1D(1:length(xw)),svector(1:length(xw)),'r'); hold on
line(onsets(1),1.2*min(svector):max(svector)/40:1.2*max(svector),'LineStyle',':')
line(onsets(1)+widths(1),1.2*min(svector):max(svector)/40:1.2*max(svector),'LineStyle',':')
ylabel('amplitude (u)');
title('Epoch 1');
axis([0 length(xw)-1 1.2*min(svector) 1.2*max(svector)]);
subplot(132)
plot(t1D(per*length(xw)+1:(per+1)*length(xw)),svector(per*length(xw)+1:(per+1)*length(xw)),'r'); hold on
line(t1D(per*length(xw))+onsets(2),1.2*min(svector):max(svector)/40:1.2*max(svector),'LineStyle','--')
line(t1D(per*length(xw))+onsets(2)+widths(2),1.2*min(svector):max(svector)/40:1.2*max(svector),'LineStyle','--')
% line(0:find(max(svector(per*length(xw)+1:(per+1)*length(xw))))
%,max(svector(per*length(xw)+1:(per+1)*length(xw))),'LineStyle','--')
set(gca,'YTickLabel',{[]})
title('Epoch 2');
axis([t1D(per*length(xw)+1) t1D((per+1)*length(xw)) 1.2*min(svector) 1.2*max(svector)]);
subplot(133)
plot(t1D(end-per*length(xw)+1:end-(per-1)*length(xw)),svector(end-per*length(xw)+1:end-(per-1)*length(xw)),'r');
line(t1D(end-per*length(xw))+onsets(N),1.2*min(svector):max(svector)/40:1.2*max(svector),'LineStyle','--')
line(t1D(end-per*length(xw))+onsets(N)+widths(N),1.2*min(svector):max(svector)/40:1.2*max(svector),'LineStyle','--')
set(gca,'YTickLabel',{[]})
set(gca,'XTickLabel',{[]})
xlabel('time (ms)');
title('Epoch N');
axis([t1D(end-per*length(xw)+1) t1D(end-(per-1)*length(xw)) 1.2*min(svector) 1.2*max(svector)]);
```

```
% Plots with Frequency Info
Y = fft(smatrix,[],2);
Ya=abs(Y);
Yp=Ya(:,1:size(Ya,2)/2);
f=linspace(0,1000*fs/2,size(Ya,2)/2);
Yp=Yp/max(max(Yp));
% 3D plot of smatrix
figure;                          % Fig. 5
surf(f,1:N,Yp)
shading flat
colormap(pink)
xlabel('frequency (Hz)');
ylabel('epoch number');
zlabel('normalized amplitude');
title(['3D Waveform=',wave,' @ ',num2str(fs),'ksps, ',num2str(nb),'-bit ADC (+/-',num2str(Amax),'u)']);
axis([0 max(f)/2.5 1 N 0 1.2*max(max(Yp))]);

figure;                           % Fig. 6
% Amplitudes and shimmer (YZ plane = FFT amplitude vs. epoch number)
subplot(221)
surf(f,1:N,Yp)
shading flat
colormap(pink)
ylabel('epoch number');
axis([0 max(f)/2.5 1 N 0 1.2*max(max(Yp))]);
set(gca,'ZTickLabel',{[]})
view(90,0)
set(gca,'PlotBoxAspectRatio',[1 1 1]);
% Waveforms (XZ plane = amplitude vs. frequency)
subplot(222)
plot(f,Yp,'Color',[207/255 169/255 146/255]); hold on
plot(f,Yp(1,:),'r','LineWidth',1); hold off  % basic
xlabel('frequency (Hz)');
```

```
ylabel('normalized amplitude');
axis([0 max(f)/2.5 0 1.2*max(max(Yp))]);
% Widths, positions and jitter (XY plane = epoch number vs. frequency)
subplot(224)
pcolor(f,1:N,Yp);
shading flat
colormap(pink)
% hold on
% plot(onsets,1:N,'r','LineWidth',1);                    % onsetss
% plot(onsets+widths,1:N,'r','LineWidth',1); hold off  % onsets+widths
ylabel('epoch number');
set(gca,'XTickLabel',{[]})
axis([0 max(f)/2.5 1 N]);
colorbar;
% Info
subplot(223)
message1=['Waveform=',wave,' @ ',num2str(fs),'ksps'];
message2=['Shimmer variation: ',aml];
message3=['Trigger variation: ',pml];
message4=['Width variation: ',wml];
text(0.05,0.8,message1);
text(0.05,0.6,message2);
text(0.05,0.4,message3);
text(0.05,0.2,message4);
set(gca,'XTickLabel',{[]})
set(gca,'YTickLabel',{[]})
axis off
```

Programa avetestnoise:

```
function [nvector,nmatrix,snr] = avetestnoise(svector,fs,nb,Amax,refm,rsa,lsa,interf,colorn,vnoise);
% Generates noise for event-related signals in a wide sense (ERSWS)
% to study signal 'averaging' methods of detection/analysis.
% Noise and interferences are generated from given amplitudes, frequencies,
% and phases for sinusoidal interferences and baseline wandering noise,
% as well as the standard deviation and cuttoff frequency, order, and
% approximation for the filter (instrumentation) to get colored noise
%
% [nvectorn,nmatrixn,snr] =
% avetestnoise(svector,fs,nb,Amax,refm,rsa,lsa,interf,colorn,vnoise);
%
% Inputs:
%   svector = column vector of ERSWS, showing the 'isoelectric' intervals
%   fs = sampling frequency, in ksamples/second, of the waveform,
%        default value is fs = 1 ksps
%   nb = number of bits of the analog to digital converter (ADC) block,
%        which determines the resolution (default value is nb=10)
%   Amax = maximun value that can be converted by the ADC
%        (default value is Amax=4000u)
%   refm = position of reference marks (fiducial marks for segmentation)
%   rsa = number of samples to the right of fid. marks, default=128
%   lsa = number of samples to the left of fid. marks, default=127
%   interf = interference and baseline wandering noise (sinusoidal-related)
%        is a matrix, in which every row (sin) comprises amp. (in u_p),
%        frequency (in Hz), and initial phase (in rads/s), for instance,
%        interf = [15, 60, pi; 7, 180, 0; 3, 300, pi/2; 10, 0.2, 0];
%   colorn = [approx, order, cutoff, stdn];
%        row vector with information of colored noise, or the
%        analog low-pass filtering (instrumentation) characteristics, ie.,
%              approx = 1 for Bessel approximation (default),
%                       2 for Butterworth approximation,
%                       3 for White Gaussian Noise (no filter)
%              order = order of the analog low-pass filtering (default=8)
%                 cutoff = cuttoff frequency in Hz (default = fs/4)
%                 stdn = standard deviation of colored noise in u
%   vnoise = vector with values of another kind of noise. This vector
%                 should be the same length of svector, otherwise it is
%                 composed by concatenation (if <), or truncated (if >)
% Outputs:
%   nvector = column vector of noisy ERSWS
%   nmatrix = N-row matrix of the noisy ERSWS, windowed for analysis
%   snr = estimated global signal to noise ratio
% Plots:
%   -3D representation of nmatrix
%   -Waveforms (XZ plane = amplitude vs. time), highlighting basic waveform
%   -Amplitudes and shimmer (YZ plane = amplitude vs. epoch number)
%   -Widths, positions and jitter (XY plane = epoch number vs. time)
%   -Plot of svector (amplitude vs. time)
%   -3D representation of nmatrix in the frequency domain
%   -Waveforms in the frequency domain (XZ plane = amplitude vs. frequency)
%   -Amplitudes and shimmer in frequency domain (YZ plane)
%   -Widths, positions and jitter in the frequency domain (XY plane)

% A. Taboada Crispi, June 2006

%--------------------------------------------------------------------------
% default values
if (nargin<1)|(isempty(svector)),
    [svector,smatrix,refm,rsa,lsa,amplits,widths,delays,fs,nb,Amax] = avetestsignal();
end
if (nargin<2)|(isempty(fs)),
    fs=1;            % default sampling frequency = 1 ksps
end
if (nargin<3)|(isempty(nb)),
    nb=10;           % default number of bits of ADC (nb = 10)
end
```

```
if (nargin<4)|(isempty(Amax)),
    Amax=4000;       % maximun value that can be converted by the ADC
end
if (nargin<5)|(isempty(refm)),
    refm=128:2*256:length(svector);     % default position of reference marks
end
if (nargin<6)|(isempty(rsa)),
    rsa=128;         % default number of samples to the right of fid. marks
end
if (nargin<7)|(isempty(lsa)),
    lsa=127;         % default number of samples to the left of fid. marks
end
%interference and baseline wandering noise (sinusoidal-related)
if (nargin<8)|(isempty(interf)),
    interf = [15, 60, pi; 7, 180, 0; 3, 300, pi/2; 10, 0.2, 0];
end
if (nargin<9)|(isempty(colorn)),
    colorn(1)=1;       % default colored noise filter approximation (Bessel)
end
if (nargin<10)|(isempty(vnoise)),
    vnoise=zeros(1,length(svector));     % default vnoise (no other noise)
end
if length(colorn)<2,
    colorn(2)=8;             % default order for coloring filter
end
if length(colorn)<3,
    colorn(3)=1000*fs/4;     % default cutoff freq for coloring filter
end
if length(colorn)<4,
    colorn(4)=25;     % default standard deviation of colored noise
end
lnoise=length(vnoise);  lvector=length(svector);
if lnoise<lvector,

    '
    k=fix(log2(lvector/lnoise));
    for i=0:k,
        vnoise=[vnoise, vnoise];
    end
end
vnoise=vnoise(1:lvector);      % complete vnoise

%------------------------------------------------------------------------
% additive noise and interference
if interf==0, interf=[0 0 0]; end
matrix=zeros(size(interf,1)+2,lvector);  % allocated matrix
for i=1:size(interf,1),                   % interference and baseline wand.
    matrix(i,:)=interf(i,1)*sin(2*pi*(0:lvector-1)*interf(i,2)/(1000*fs)+interf(i,3));
end
matrix(size(interf,1)+1,:)=vnoise;              % other noise
% colored noise, colorn = [approx, order, cutoff, stdn];
if colorn(1) == 1,                              % if Bessel filter
    [Ba,Aa]=besself(colorn(2),2*pi*colorn(3));  % analog coefficients
    [B,A]=impinvar(Ba,Aa,fs*1000);              % equiv digital coeficients
elseif colorn(1) == 2,                          % if Butterworth filter
    [B,A]=butter(colorn(2),colorn(3)/(1000*fs/2));% digital coefficients
else
    B=1; A=1;                                   % if White Gaussian noise
end
tmp=filter(B,A,randn(1,lvector));
matrix(size(interf,1)+2,:)=colorn(4)*tmp/std(tmp);    % colored noise
nvector=sum(matrix);                           % total additive noise
nvector=nvector-mean(nvector);                 % takes DC level off
nvector=nvector+svector;                       % noisy vector
%------------------------------------------------------------------------
% ADC quantization (not affecting first epoch)
lsb=Amax/2^(nb-1);
nvector(rsa+lsa+1:end)=lsb*round(nvector(rsa+lsa+1:end)/lsb);
```

```
nvector(nvector>Amax)=Amax;                % ADC range limitation
nvector(nvector<-Amax)=-Amax;
%-------------------------------------------------------------------------
% Noisy matrix
nmatrix=windowingmatrix(nvector,refm,rsa,lsa);
%-------------------------------------------------------------------------
snr=10*log10(var(svector)/var(nvector));  % signal to noise ratio
%-------------------------------------------------------------------------
% PLOTS
t=0:1/fs:size(nmatrix,2)-1/fs;      % time axis (ms) for nmatrix
t1D=0:1/fs:lvector-1;               % time axis (ms) for svector
N=size(nmatrix,1);
per=lvector/((rsa+lsa+1)*N);
% 3D plot of smatrix
figure;                          % Fig. 1
surf(t,1:N,nmatrix)
shading flat
% colormap(gray)
colormap(pink)
xlabel('time (ms)');
ylabel('epoch number');
zlabel('amplitude (u)');
title(['3D Noisy Waveform @ ',num2str(fs),'ksps, ',num2str(nb),'-bit ADC (+/-',num2str(Amax),'u)']);
axis([0 max(t) 1 N 1.2*min(svector) 1.2*max(svector)]);

figure;                           % Fig. 2
% Amplitudes and shimmer (YZ plane = amplitude vs. epoch number)
subplot(221)
plot(1:N,nmatrix','Color',[207/255 169/255 146/255]); hold on
plot(1:N,max(nmatrix'),'r','LineWidth',2); hold off % shimmer
xlabel('epoch number');
set(gca,'YTickLabel',{[]})
```

```
% Waveforms (XZ plane = amplitude vs. time)
subplot(222)
plot(t,nmatrix,'Color',[207/255 169/255 146/255]); hold on
plot(t,svector(1,1:rsa+lsa+1),'r','LineWidth',1); hold off   % basic
xlabel('time (ms)');
ylabel('amplitude (u)');
axis([0 max(t) 1.2*min(svector) 1.2*max(svector)]);
% Widths, positions and jitter (XY plane = epoch number vs. time)
subplot(224)
pcolor(t,1:N,nmatrix);
shading flat
% colormap(gray)
colormap(pink)
% hold on
% plot(onsets,1:N,'r','LineWidth',1);                    % onsetss
% plot(onsets+widths,1:N,'r','LineWidth',1); hold off   % onsets+widths
ylabel('epoch number');
set(gca,'XTickLabel',{[]})
axis([0 max(t) 1 N]);
colorbar;
% Info
subplot(223)
message1=['Noisy Waveform @ ',num2str(fs),'ksps'];
text(0.05,0.5,message1);
set(gca,'XTickLabel',{[]})
set(gca,'YTickLabel',{[]})
axis off

% Plot of nvector (amplitude vs. time)
figure;                              % Fig. 3
plot(t1D,nvector,'r'); grid
xlabel('time (ms)');
ylabel('amplitude (u)');
```

```
title(['1D Noisy Waveform @ ',num2str(fs),'ksps, ',num2str(nb),'-bit ADC (+/-',num2str(Amax),'u)']);
axis([0 max(t1D) 1.2*min(svector) 1.2*max(svector)]);

% Plot of nvector (amplitude vs. time) for epochs 1, 2, and N
figure;                         % Fig. 4
subplot(131)
plot(t1D(refm(1)-lsa:refm(1)+rsa),nvector(refm(1)-lsa:refm(1)+rsa),'r');
% hold on
% line(onsets(1),1.2*min(rvector):max(rvector)/40:1.2*max(rvector),'LineStyle',':')
% line(onsets(1)+widths(1),1.2*min(rvector):max(rvector)/40:1.2*max(rvector),'LineStyle',':')
ylabel('amplitude (u)');
title('Epoch 1');
axis([t1D(refm(1)-lsa) t1D(refm(1)+rsa) 1.2*min(nvector) 1.2*max(nvector)]);
subplot(132)
plot(t1D(refm(2)-lsa:refm(2)+rsa),nvector(refm(2)-lsa:refm(2)+rsa),'r');
% hold on
% line(t1D(per*(rsa+lsa+1))+onsets(2),1.2*min(rvector):max(rvector)/40:1.2*max(rvector),'LineStyle','--')
% line(t1D(per*(rsa+lsa+1))+onsets(2)+widths(2),1.2*min(rvector):max(rvector)/40:1.2*max(rvector),'LineStyle','--')
% line(0:find(max(rvector(per*length(xw)+1:(per+1)*length(xw))))
%,max(rvector(per*length(xw)+1:(per+1)*length(xw))),'LineStyle','--')
set(gca,'YTickLabel',{[]})
title('Epoch 2');
axis([t1D(refm(2)-lsa) t1D(refm(2)+rsa) 1.2*min(nvector) 1.2*max(nvector)]);
subplot(133)
plot(t1D(refm(N)-lsa:refm(N)+rsa),nvector(refm(N)-lsa:refm(N)+rsa),'r');
% line(t1D(end-per*length(xw))+onsets(N),1.2*min(rvector):max(rvector)/40:1.2*max(rvector),'LineStyle','--')
% line(t1D(end-per*length(xw))+onsets(N)+widths(N),1.2*min(rvector):max(rvector)/40:1.2*max(rvector),'LineStyle','--')
set(gca,'YTickLabel',{[]})
set(gca,'XTickLabel',{[]})
xlabel('time (ms)');
title('Epoch N');
axis([t1D(refm(N)-lsa) t1D(refm(N)+rsa) 1.2*min(nvector) 1.2*max(nvector)]);
```

```
% Plots with Frequency Info
Y = fft(nmatrix,[],2);
Ya=abs(Y);
Yp=Ya(:,1:size(Ya,2)/2);
Yi=fft(svector(1:rsa+lsa+1));
Yai=abs(Yi);
Ypi=Yai(:,1:length(Yai)/2);
f=linspace(0,1000*fs/2,size(Ya,2)/2);

% 3D plot of smatrix
figure;                             % Fig. 5
surf(f,1:N,Yp)
shading flat
colormap(pink)
xlabel('frequency (Hz)');
ylabel('epoch number');
zlabel('amplitude (u)');
title(['3D Noisy Waveform @ ',num2str(fs),'ksps, ',num2str(nb),'-bit ADC (+/-',num2str(Amax),'u)']);
axis([0 max(f)/2.5 1 N 0 1.2*max(max(Yp))]);

figure;                              % Fig. 6
% Amplitudes and shimmer (YZ plane = FFT amplitude vs. epoch number)
subplot(221)
plot(1:N,Yp','Color',[207/255 169/255 146/255]);
% hold on
% plot(1:N,max(smatrix'),'r','LineWidth',2); hold off % shimmer
xlabel('epoch number');
set(gca,'YTickLabel',{[]})
axis([1 N 0 1.2*max(max(Yp))]);
set(gca,'PlotBoxAspectRatio',[1 1 1]);
% Waveforms (XZ plane = amplitude vs. frequency)
subplot(222)
```

```
plot(f,Yp,'Color',[207/255 169/255 146/255]); hold on
plot(f,Ypi,'r','LineWidth',1); hold off  % basic
xlabel('frequency (Hz)');
ylabel('amplitude (u)');
axis([0 max(f)/2.5 0 1.2*max(max(Yp))]);
% Widths, positions and jitter (XY plane = epoch number vs. frequency)
subplot(224)
pcolor(f,1:N,Yp);
shading flat
colormap(pink)
% hold on
% plot(onsets,1:N,'r','LineWidth',1);                    % onsetss
% plot(onsets+widths,1:N,'r','LineWidth',1); hold off  % onsets+widths
ylabel('epoch number');
set(gca,'XTickLabel',{[]})
axis([0 max(f)/2.5 1 N]);
colorbar;
% Info
subplot(223)
message1=['Noisy Waveform @ ',num2str(fs),'ksps'];
text(0.05,0.5,message1);
set(gca,'XTickLabel',{[]})
set(gca,'YTickLabel',{[]})
axis off
```

Programa ccrcoef:

```
function [ccr]=ccrcoef(fig,xmin,xmax,ymax,fs)
%ccrcoef is a funcion who first add a noise (at 60hz and 180hz of
%amplitud 0.4)at a given signal(image of the signal)
%and then calculate the crr values using 4 methods (Ensembe mean,Trimeed
%mean,modified trimmed mean,median filter)
%FORMAT:
%[ccr]=ccrcoef(fig,xmin,xmax,ymax,fs);
%INPUTS:
%fig= the image of the evoked potencial signal.
%xmin=minumum value (Time)of the signal.
%xmax=maximum value (Time)of the signal.
%fs=sample frequency.
%ymax=the maximun amplitud u consider that the signal has.
%OUTPUTS:
%ccr=a vector  of values of ccr using the 4 methods .
[A,t]=graph2vector(fig,xmin,xmax,ymax,fs);
[svector,smatrix,refm,rsa,lsa,amplits,widths,delays,fs,nb,Amax] = avetestsignal('othr',A,8);
pause;
[nvectorn,nmatrixn,snr] = avetestnoise(svector,fs,nb,Amax,refm,rsa,lsa,[.4 60 0],[1 8 240 .4]);
pause;
refm1=refm(1:2:end);
refm2=refm(2:2:end);
[ya1,ym1] = ens_ave(nvectorn,refm1,rsa,lsa,1,64,0,0);

[ya2,ym2] = ens_ave(nvectorn,refm2,rsa,lsa,1,64,0,0);

 CCR1=corrcoef(ya1,ya2)
 [ya3,ym3] = ens_ave(nvectorn,refm1,rsa,lsa,1,64,1,0);

[ya4,ym4] = ens_ave(nvectorn,refm2,rsa,lsa,1,64,1,0);
```

```
 CCR2=corrcoef(ya3,ya4)
 [ya5,ym5] = ens_ave(nvectorn,refm1,rsa,lsa,1,64,2,0);

[ya6,ym6] = ens_ave(nvectorn,refm2,rsa,lsa,1,64,2,0);
 CCR3=corrcoef(ya5,ya6)
 [ya7,ym7] = ens_ave(nvectorn,refm1,rsa,lsa,1,64,3,0);

[ya8,ym8] = ens_ave(nvectorn,refm2,rsa,lsa,1,64,3,0);
 CCR4=corrcoef(ya7,ya8)
```

Programa ens_ave:

```
function [ya,ym,d] = ens_ave(ns,refm,rsa,lsa,sc,n,t,ln,d)
%  Ensemble averaging: obtains the averaging of some segments
%  around (+rsa/-lsa) the fiducial marks (refm) in x:
%  n segments if sc==1, otherwise untill noise be low enough
%
% Format:
%      [ya,ym] = ens_ave(ns,refm,rsa,lsa,sc,n,t,ln)
%
% Arguments:
%     ns = input noisy signal
%   refm = position of reference marks in ns (fid. marks for segment.)
%    rsa = number of samples to the right of fid. marks in segment, default=156
%    lsa = number of samples to the left of fid. marks in segment, default=99
%     sc = stop acquisition criteria
%          1. number of beats, default
%          2. noise level
%      n = number of beats to average, if sc==1 (n<length(refm)), default,
%          or noise level in microVolts, if sc==2, to stop the ave.
%      t = averaging method
%          0. ensemble mean (mean of the whole ensemble)
%          1. trimmed mean (the bottom 5% and top 5% values are discarded), default
%          2. modified trimmed mean (see modtrim help)
%          3. median filter
%     ln = lead normalization
%          0. no lead normalization to supress respiration effects (amplitude modulation and baseline wand.)
%          1. lead normalization to supress respiration effects (default)
%
% Returns:
%      ya = column vector, output signal (ensemble average)
%      ym = matrix (before trimming and averaging)

temp=size(ns);
if nargin<8, ln=1; end
```

```
if nargin<7, t=2; end
if nargin<6, n=length(refm); end
n=60;%*****************************************
if nargin<5, sc=1; end
if nargin<4, lsa=99; end
if nargin<3, rsa=156; end
if temp(1)>temp(2), ns=ns'; end
ri=40;                                          % determines right isopot. segment
li=40;                                          % determines left isopot. segment
if sc==1,
    n=min(n,length(refm));
    x=zeros(n,rsa+lsa+1);
    for i=1:n,
        %x(i,:)=ns(refm(i)-lsa:refm(i)+rsa)-mean(ns(refm(i)-lsa:refm(i)+rsa)); % removes DC level from every segment
        x(i,:)=ns(refm(i)-lsa:refm(i)+rsa);        % windowing
   end
   if ln==1, x=lead_normalizing(x); end             % lead normalization to supress respiration effects
   ym=x;
   if n>1,
        if t==0,                                    % mean filter (coherent averaging)
            y=mean(x);
        elseif t==1,                                % trimmed mean
            x=sort(x);                              % sort values at every sample
            x=x(round(0.05*n):n-round(0.05*n),:);   % discards bottom 5% and top 5%
            y=mean(x);
        elseif t==2,                                % modified trimmed mean
%*********************************
            %[y,d]=modtrim(x,12.6,d); % for dmin_study
            %y=modtrim(x,12.6);     % for q_study
            y=modtrim(x);               % normal
        elseif t==3,                                % median filter
            y=median(x);
        end
```

```
    else,
        y=x;
    end


elseif sc==2,                                       % *** OBSOLETE *** (HAVE TO BE UPDATED) ***
    x=[]; no=100;
    x(1,:)=ns(refm(1)-lsa:refm(1)+rsa)-mean(ns(refm(1)-lsa:refm(1)+rsa));
    i=2;
    while (no>n)&(i<length(refm)),
        x(i,:)=ns(refm(i)-lsa:refm(i)+rsa)-mean(ns(refm(i)-lsa:refm(i)+rsa));
      if t==1,                                              % trimmed mean
        xt=sort(x);                                 % sort values at every sample
         xt=x(round(0.05*n):n-round(0.05*n),:); % discards bottom 5% and top 5%
      else,
        xt=x;
       end
       y=mean(xt);
        no=[y(1:li)   y(rsa+lsa+1-ri:rsa+lsa+1)];% isopot segment (right-left)
        no=sqrt(mean((no-mean(no)).^2));        % Vnoise (rms) in isopot segm.
        i=i+1;
    end
end
ya=y;
```

Programa modtrim:

```
function [y,d,m,w]=modtrim(x,stdiso,d,q);
% Modified trimmed mean for VLP detection
% Format:
%       [y,m,w]=modtrim(x,d,stdiso);
% Inputs:
%       x = input matrix (heartbeats,samples)
%  stdiso= standard deviation of the isoelectric segment. It can be automaticaly estimated if not given.
%       d = number of displacements taken into account. It can be automaticaly estimated if not given.
%       q = trimming parameter. It can be automaticaly estimated if not given.
% Outputs:
%       y = trimmed mean output vector
%       m = intermediate matrix to compute the mean
%       w = weighting matrix to compute the mean


k=0.4;
noiseout=0.7;                           % output noise to estimate d if not given
[N,c]=size(x);
if N==1,
    y=x;
    return
end

if (nargin<2)|isempty(stdiso),
    stdt=zeros(1,c-100);                % stdiso estimation (from 100sample segments)
    for i=1:c-100,
        stdt(i)=std(x(1,i:i+100));
    end
    stdiso=min(stdt);
    stdiso=max(stdiso,1);             % min stdiso expected is 1
end
```

```
    end
    stdiso=min(stdt);
    stdiso=max(stdiso,1);               % min stdiso expected is 1
end
if nargin<3|isempty(d),
    %d=fix((((stdiso/(k*noiseout))^2)/N-1)/2)+1;
    %d=max(d,1);                         % min d is 1
    d=10;
end

if nargin<4, q=2*stdiso; end            % trimming parameter

med=median1(x);
m=zeros((2*d+1)*N,c);                   % allocated matrix
for i=1:(2*d+1)*N,
    m(i,:)=med;                         % default columns are median column
end
m(1:N,:)=x;
for i=1:d,
    m(N+2*N*(i-1)+1:2*N*i,1:c-i)=x(:,i+1:c);    % i-th advanced version
    m(2*N*i+1:(2*i+1)*N,i+1:c)=x(:,1:c-i);      % i-th delayed version
end
[r,c]=size(m);
tmp=zeros(r,c);                         % allocated matrix
for i=1:r,
    tmp(i,:)=abs(m(i,:)-med);
end
w=ones(r,c);                            % weighting matrix
w(find(tmp>q))=0;                     % wkj=0 if mkj differs more than q from median(xj)
y=sum(w.*m)./sum(w);                    % trimmed mean output vector
```

Programa graph2vector:

```
function [w,t]=graph2vector(fig,xmin,xmax,ymax,fs)
%graph2vector generates a pair of vectors t,a
%from a graphical representation of a waveform
%limited between Xmin and Xmax,and given the sampling
%frequency (fs)and the maximum point that you wish to have the signal
%and the figure also ,remember here that when you are going to put the
%inputs that you have to put the format of the image for example,image.bmp.
%FORMAT:
%[A,t]=graph2vector(fig,xmin,xmax,ymax,fs);
%INPUTS:
%fig= the image of the evoked potencial signal.
%xmin=minumum value (Time)of the signal.
%xmax=maximum value (Time)of the signal.
%fs=sample frequency.
%ymax=the maximun amplitud u consider that the signal has.
%OUTPUTS:
%A=a vector amplitud of the signal.
%t=a vector time.
fs=round(fs);
tmp = imread(fig);
tmp=tmp(:,:,1);
[m,n,p] = size(tmp);
fs1=round(n/(xmax-xmin));
y = zeros(1,m);
for i = 1:n,
    for j = m:-1:1
        if tmp(j,i)< 50
            y(i) = j;
            break;
        end;
    end;
end
y = -y;
y=y-mean(y);
z=resample(y,fs,fs1);
```

```
w=ymax*z/max(z);
t=linspace(xmin,xmax,length(w));
plot(t,w)
xlabel('Time (ms)')
ylabel('Amplitud (microvoltio)')
if min(w)>0, miny=0.9*min(w); else, miny=1.1*min(w); end
if max(w)<0, maxy=0.9*max(w); else, maxy=1.1*max(w); end
axis([min(t) max(t) miny maxy])
```

Programa plotvector:

```
function =plot2vector(m,fs)
%plotvector generate a graph of a given signal with its sample
%frequency.
%FORMAT:
%plotvector(m,fs);
%INPUTS:
%m es el vector de la senal promediada.
%fs= es la frecuencia de muestreo con la cual se hizo el proceso de muestreo.
%OUTPUTS:
%-representation of the signal (amplitud vs Time).
c=length(m);% cantidad de muestras de la senal.
t=fs*c;%el tiempo completo de la prueba.
ts=1/fs;%periodo de muestreo.
j=ts;
x=zeros(c,1);
for i=1:c
    x(i)=j;
   j=j+ts;
end
yy=spline(x,m,linspace(x(1),x(end),c));%senal recuperada mediante spline.
length(yy)
tt=0:ts:(t-1);
length(tt)
plot(tt,-yy)
```

Programa filtrado:

```
function [mejores]=filtrado(fig,xmin,xmax,ymax,fs,fc)
%filtrado is a funcion who first add a noise at 60Hz, and also at
%180Hz of amplitud 0.4 at a given signal(an image of a signal)and then  filter it
%by filters bessel,butterworth first orden ,butterworth second orden
%,chebychev type 1 order 1 and having a 30db as an atenuation value
%at the value of cutoff frequecy(all chebychev filters has the same
%atenuation value at the cutoff frequency),chebychev type 2 order 1 and
%,chebychev type 2 order 1 ,chebychev
% type 2 order 2 ,given a figure of a signal and the xmin ,xmax,ymax,fs,fc
%and these are explained down in inputs.
%FORMAT:
%[mejores]=filtrado(fig,xmin,xmax,ymax,fs,fc);
%INPUTS:
%fig= the image of the evoked potencial signal.
%xmin=minumum value (Time)of the signal.
%xmax=maximum value (Time)of the signal.
%fs=sample frequency.
%ymax=the maximun amplitud u consider that the signal has.
%fc=cutoff frequency.
%OUTPUTS:
%mejores=is a vector who returns values of the SNR of each filter
%if they are better than the original one ,if not ,it will have a zero value.
 %Plots:
 %-representation of each signal after it pass by any filter.
[A,t]=graph2vector(fig,xmin,xmax,ymax,fs);
pause;
[svector,smatrix,refm,rsa,lsa,amplits,widths,delays,fs,nb,Amax] = avetestsignal('othr',A,ymax);
pause;
[nvectorn,nmatrixn,snr] = avetestnoise(svector,fs,nb,Amax,refm,rsa,lsa,[.4 60 0],[1 8 180 .4]);
pause;
potenciaoriginal=var(svector);
ruido=nvectorn-svector;
plot(ruido)

title('Addicional noise')
xlabel('Time (ms)')
ylabel('Amplitud (microvoltio)')
pause;
SDR1=std(svector)/std(ruido)
potenciaruido=var(ruido);
RSR=10*log(potenciaoriginal/potenciaruido)
plot(svector)
title(' Signal without noise')
xlabel('Time (ms)')
ylabel('Amplitud (microvoltio)')
pause;
plot(nvectorn)
title('Signal with noise')
xlabel('Time (ms)')
ylabel('Amplitud (microvoltio)')
pause;
[b a]=besself(20,fc,'low');
y=filtfilt(b,a,nvectorn);
plot(y)
title('Signal filtring with bessel low passs filter of orden 20 and cutoff frequency .04(20Hz)')
xlabel('Time (ms)')
ylabel('Amplitud (microvoltio)')
pause;
s1=y-svector;
potencia20=var(s1);
RSR20=10*log(potenciaoriginal/potencia20)
[b a]=butter(2,fc,'low');
y=filtfilt(b,a,nvectorn);
plot(y)
title('Signal filtring with butterworth low passs filter of orden 2 and cutoff frequency .04(20Hz)')
xlabel('Time (ms)')
ylabel('Amplitud (microvoltio)')
```

```
pause;
SDR2=std(y)/std(ruido)
s2=y-svector;
potencia21=var(s2);
RSR21=10*log(potenciaoriginal/potencia21)
pause;
[b a]=butter(1,fc,'low');
y=filtfilt(b,a,nvectorn);
plot(y)
title('Signal filtring with butterworth low passs filter of orden 1 and cutoff frequency .04(20Hz)')
xlabel('Time (ms)')
ylabel('Amplitud (microvoltio)')
pause;
s3=y-svector;
potencia22=var(s3);
RSR22=10*log(potenciaoriginal/potencia22)
pause;
[b a]=cheby1(1,30,fc,'low');
y=filtfilt(b,a,nvectorn);
plot(y)
title('Signal filtring with chebychev low passs filter(type 1) of orden 1 and cutoff frequency .04(20Hz)')
xlabel('Time (ms)')
ylabel('Amplitud (microvoltio)')
pause;
s5=y-svector;
potencia24=var(s5);
RSR24=10*log(potenciaoriginal/potencia24)
pause;
[b a]=cheby2(1,30,fc,'low');
y=filtfilt(b,a,nvectorn);
plot(y)
title('Signal filtring with chebychev low passs filter(type 2) of orden 1 and cutoff frequency .04(20Hz)')
xlabel('Time (ms)')
```

```
ylabel('Amplitud (microvoltio)')
pause;
s6=y-svector;
potencia25=var(s6);
RSR25=10*log(potenciaoriginal/potencia25)
pause;
[b a]=cheby1(2,30,fc,'low');
y=filtfilt(b,a,nvectorn);
plot(y)
title('Signal filtring with chebychev low passs filter(type 1) of orden 2 and cutoff frequency .04(20Hz)')
xlabel('Time (ms)')
ylabel('Amplitud (microvoltio)')
pause;
s7=y-svector;
potencia26=var(s7);
RSR26=10*log(potenciaoriginal/potencia26)
pause;
[b a]=cheby2(2,30,fc,'low');
y=filtfilt(b,a,nvectorn);
plot(y)
title('ASignal filtring with chebychev low passs filter(type 2) of orden 2 and cutoff frequency .04(20Hz)')
xlabel('Time (ms)')
ylabel('Amplitud (microvoltio)')
pause;
s8=y-svector;
potencia27=var(s8);
RSR27=10*log(potenciaoriginal/potencia27)
pause;
rsr=[RSR20 RSR21 RSR22 RSR24 RSR25 RSR26 RSR27];
mejores = zeros(1,7);
for i=1:7
    if rsr(i)>RSR
        mejores(i)=rsr(i);
```

```
    end;
end;
mejores
```

Programa leadnormalizing:

```
function [os,xn,xn1,xn2,xn3,xn4,AM1,AM2,AM3,AM4,mea,DC,ini,fin]=leadnormalizing(x,refm,rsa,lsa,winc
% normalizes lead data to attenuate respiration effects
% (baseline wandering and amplitude modulation)
% Format:
%[os,xm] = leadnormalizing(x,refm,rsa,lsa,window);
% Input:
%       x = input vector
%    refm = position of reference marks in os (fid. marks for window.)
%     rsa = number of samples to the right of fid. marks
%     lsa = number of samples to the left of fid. marks
% window = window type (see window help)
%
% Returns:
%      os = output vector obtained by windowing (normalized)
%      xn = matrix output (normalized)
%     ini = first sample of the 50ms reference isosegment (OV)
%     fin = final sample of the 50ms reference isosegment (OV)
%
%-------------------------------------------------------------------------

fs=1;
if nargin<5,
   window=boxcar(rsa+lsa+1);
end

[r,c]=size(x);
if r>c, x=x'; end
refm=refm(refm~=0);                          % remove terms=0 from refm
k=length(refm);                              % number of segments (beats)
os=zeros(1,k*(rsa+lsa+1));
xn=zeros(k,rsa+lsa+1);
pk=zeros(k,1);
[B,A]=butter(4,2/(fs*1000/2),'high');   % HPF
fx=filtfilt(B,A,x);

for n=1:k,                                           % for each segment
    xn(n,:)=x(refm(n)-lsa:refm(n)+rsa);
    fxn(n,:)=fx(refm(n)-lsa:refm(n)+rsa);
    %pk(n,:)=mean(fx(refm(n)-2:refm(n)+2));
end

xave=mean(fxn);                                      % raw averaging
%pkave=mean(xave(98*fs:fs*102));
%p=polyfit(refm(1:k),pk',2);

nstd=zeros(1,length(xave)-round(50*fs));
for  i=1:length(xave)-round(50*fs),
     nstd(i)=std(xave(i:i+round(50*fs)-1));   % std of the average signal
end
noise=min(nstd);
ini=min(find(nstd==noise));
fin=ini+round(50*fs)-1;

Pxave=mean((xave-mean(xave(ini:fin))).^2)-var(xave(ini:fin));
Vxave=var(xave)-var(xave(ini:fin));
Pkave=max(xave-mean(xave(ini:fin)));
distave=prctile(xave,75)-prctile(xave,25);

DC=zeros(1,k);
AM1=zeros(1,k);
AM2=zeros(1,k);
AM3=zeros(1,k);
AM4=zeros(1,k);
mea=zeros(1,k);
for n=1:k,                                           % for each segment
    DC(n)=mean(xn(n,ini:fin));
    mea(n)=mean(xn(n,:));
    xO=xn(n,:)-mean(xn(n,ini:fin));
    xm=xn(n,:)-mean(xn(n,:));
```

```
    Px=mean(x0.^2)-var(xn(n,ini:fin));
    Vx=var(xn(n,:))-var(xn(n,ini:fin));
     Pk=max(x0);
     dist=prctile(xn(n,:),75)-prctile(xn(n,:),25);
     xn1(n,:)=(x0*Pxave/Px).*window';
    xn2(n,:)=(x0*Vxave/Vx).*window';
    xn3(n,:)=(x0*Pkave/Pk).*window';
    xn4(n,:)=(xm*distave/dist).*window';
    os(1,(n-1)*(rsa+lsa+1)+1:n*(rsa+lsa+1))=xn(n,:);
    AM1(n)=Pxave/Px;
    AM2(n)=Vxave/Vx;
    AM3(n)=Pkave/Pk;
    AM4(n)=distave/dist;
end
```

Programa sdrcoef:

```
function [sdr,potencia]=sdrcoef(fig,xmin,xmax,ymax,fs);
%sdrcoef  generates a pair of vectors sdr,potencia
%from a graphical representation of a waveform
%limited between Xmin and Xmax,and given the sampling
%frequency (fs)and the maximum point that you wish to have the signal
%and the figure also ,remember here that when you are going to put the
%inputs that you have to put the format of the image for example,image.bmp.
%also we say that sdr coef add a noise (at 60 and 180Hz) at a given signal (image).
%FORMAT:
%[sdr,potencia]=sdrcoef(fig,xmin,xmax,ymax,fs);
%INPUTS:
%fig= the image of the evoked potencial signal.
%xmin=minumum value (Time)of the signal.
%xmax=maximum value (Time)of the signal.
%fs=sample frequency.
%ymax=the maximun amplitud you consider that the signal has.
%OUTPUTS:
%sdr=a vector that has 4 values of sdr by orden for each type
%of avereging methods(ensemble mean,timmed mean,modified timmed mean,median mean).
%potencia=a vector that has also 4 values of power noise by orden for
%each type of avereging methods(ensemble mean,timmed mean,modified timmed mean,median mean)..
[A,t]=graph2vector(fig,xmin,xmax,ymax,fs);
 [svector,smatrix,refm,rsa,lsa,amplits,widths,delays,fs,nb,Amax] = avetestsignal('othr',A,8);
 close all;
 [nvectorn,nmatrixn,snr] = avetestnoise(svector,fs,nb,Amax,refm,rsa,lsa,[.4 60 0],[1 8 180 .4]);
close all;
%para calcular sdr
 idealepoch=svector(refm(1)-lsa:refm(1)+rsa);
 plot(idealepoch)
 %usando el primer promediado
 [ya1,ym1] = ens_ave(nvectorn,refm,rsa,lsa,1,64,0,0);
 noise=ya1-idealepoch;
```

```
pause;
plot(noise)
potencia1=var(noise);
sdr1=std(idealepoch)/std(noise);
%usando el segundo promediado
[ya2,ym2] = ens_ave(nvectorn,refm,rsa,lsa,1,64,1,0);
noise=ya2-idealepoch;
pause;
plot(noise)
potencia2=var(noise);
sdr2=std(idealepoch)/std(noise);
%usando el tercer promediado
[ya3,ym3] = ens_ave(nvectorn,refm,rsa,lsa,1,64,2,0);
noise=ya3-idealepoch;
pause;
plot(noise)
potencia3=var(noise);
sdr3=std(idealepoch)/std(noise);
%usando el 4to promediado
[ya4,ym4] = ens_ave(nvectorn,refm,rsa,lsa,1,64,3,0);
noise=ya4-idealepoch;
pause;
plot(noise)
potencia4=var(noise);
sdr4=std(idealepoch)/std(noise);
sdr=[sdr1 sdr2 sdr3 sdr4];
potencia=[potencia1 potencia2 potencia3 potencia4];
```

Programa nrrcoef:

```
function [nrr]=nrrcoef(fig,xmin,xmax,ymax,fs);
%nrrcoef  generates  vector nrr
%from a graphical representation of a waveform
%limited between Xmin and Xmax,and given the sampling
%frequency (fs)and the maximum point that you wish to have the signal
%and the figure also ,remember here that when you are going to put the
%inputs that you have to put the format of the image for example,image.bmp.
%also we say that nrrcoef add a noise (at 60 and 180Hz) at a given signal (image).
%FORMAT:
%[nrr]=nrrcoef(fig,xmin,xmax,ymax,fs);
%INPUTS:
%fig= the image of the evoked potencial signal.
%xmin=minumum value (Time)of the signal.
%xmax=maximum value (Time)of the signal.
%fs=sample frequency.
%ymax=the maximun amplitud you consider that the signal has.
%OUTPUTS:
%nrr=a vector that has 4 values of nrr by orden for each type
%of avereging methods(ensemble mean,timmed mean,modified timmed mean,median
%mean).
[A,t]=graph2vector(fig,xmin,xmax,ymax,fs);
 [svector,smatrix,refm,rsa,lsa,amplits,widths,delays,fs,nb,Amax] = avetestsignal('othr',A,ymax);
 close all;
 [nvectorn,nmatrixn,snr] = avetestnoise(svector,fs,nb,Amax,refm,rsa,lsa,[.4 60 0],[1 8 180 .4]);
close all;
%para calcular el factor nrr mediante el primer tipo de promediado(ensemble mean)
 [ya1,ym1] = ens_ave(nvectorn,refm,rsa,lsa,1,64,0,0);
 [m,n]=size(ym1);
 ytmp1=ym1;
 ytmp1(2:2:m,:)=-ytmp1(2:2:m,:);
 nnr1=std(sum(ytmp1));

 %para calcular el factor nrr mediante el segundo tipo de promediado(trimmed men)
 [ya2,ym2] = ens_ave(nvectorn,refm,rsa,lsa,1,64,1,0);
 [m,n]=size(ym2);
 ytmp2=ym2;
 ytmp2(2:2:m,:)=-ytmp2(2:2:m,:);
 nnr2=std(sum(ytmp2));
 %para calcular el factor nrr mediante el tercer tipo de promediado(modified trimmed mean)
 [ya3,ym3] = ens_ave(nvectorn,refm,rsa,lsa,1,64,2,0);
 [m,n]=size(ym3);
 ytmp3=ym3;
 ytmp3(2:2:m,:)=-ytmp3(2:2:m,:);
 nnr3=std(sum(ytmp3));
  %para calcular el factor nrr mediante el cuarto tipo de promediado(median filter)
 [ya4,ym4] = ens_ave(nvectorn,refm,rsa,lsa,1,64,3,0);
 [m,n]=size(ym4);
 ytmp4=ym4;
 ytmp4(2:2:m,:)=-ytmp4(2:2:m,:);
 nnr4=std(sum(ytmp4));
```