Procesamiento Digital de Señales de Audio

Práctico 1

Santiago García Pose - 4.595.400-6

30 de marzo de 2020

Índice

1. Problema 1 - Muestreo y cuantización **3**
1.1. Parte 1 - Muestreo . . . 3
1.1.1. Archivo `tones.wav` . . . 3
1.1.2. Archivo `chirp.wav` . . . 4
1.2. Parte 2 . . . 5
1.2.1. 1) . . . 5
1.2.2. 2) . . . 7
1.2.3. 3) . . . 8

2. Problema 2 - Características en tiempo corto de una señal de audio **9**
2.1. Parte 1 . . . 9
2.1.1. 1) . . . 9
2.2. Parte 2 . . . 12
2.2.1. 1) . . . 12
2.2.2. Detección de palabras utilizando la energía y tasa de cruces por cero. 13

3. Problema 3 - Cálculo y aplicación de la función de autocorrelación **16**
3.1. Parte 1 - Propiedades de la autocorrelación . . . 16
3.1.1. 1) . . . 16
3.1.2. 2) . . . 16
3.2. Parte 2 - Estimación de la frecuencia fundamental . . . 17

1. Problema 1 - Muestreo y cuantización

1.1. Parte 1 - Muestreo

1.1.1. Archivo `tones.wav`

Para esta parte se trabajó con dos tonos puros muestreados a 44100 Hz de frecuencias 1760 Hz y 7040 Hz. En primera instancia se re-muestrearon a una frecuencia de 22050 Hz, y en segunda instancia se realizó otro re-muestreo a una frecuencia de 11025 Hz, es decir, se dividió a la mitad la frecuencia de sampleo dos veces. Ambos procedimientos se realizaron por un lado descartando una de cada dos muestras en la señal, y por otro utilizando la función `decimate` del paquete `scipy.signal`. Las observaciones se realizan a continuación.

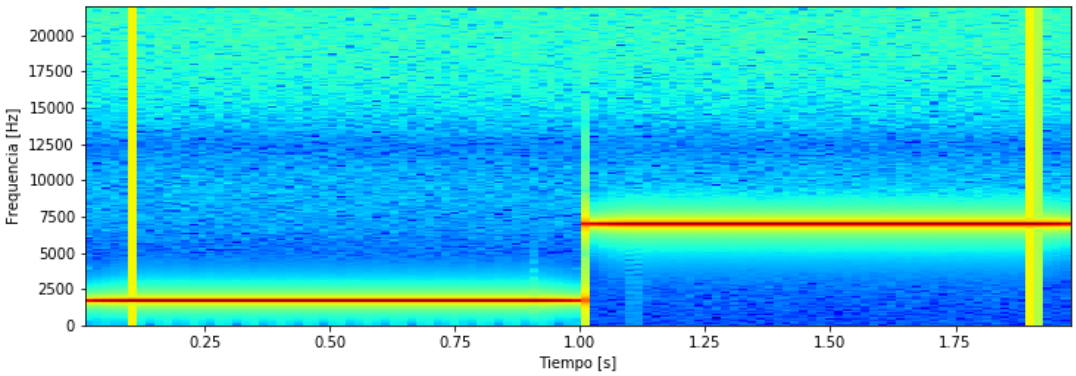


Figura 1.1: Espectrograma de los tonos originales correspondientes a 1760 Hz y 7040 Hz.

Descartando una cada dos muestras

Para el primer re-muestreo se pueden notar sonoramente dos cosas, la primera es que los tonos permanecen en sus respectivas alturas, sin embargo el pequeño 'crackle' que existía (en el comienzo, el cambio de tono y al final) deja de ser tan notorio. Esto se puede además comprobar visualmente en el espectrograma, las barras verticales de energía se hacen menos intensas.

En el caso del segundo re-muestreo, es decir, a 11025 Hz, los resultados ya son un poco distintos y se apartan de lo visto en el primer re-muestreo. Como se puede apreciar en la figura 1.2 en comparación con la figura 1.1, las barras verticales que correspondían a esa especie de 'crackle' que eran perfectamente audibles desaparecen, tanto gráficamente como de forma auditiva.

Pero lo más notorio, es que observamos un claro problema de *aliasing*, que se debe a que el segundo tono excede la frecuencia de Nyquist (en este caso 5512,5 Hz), por lo que escuchamos producto de esto, un tono con frecuencia teórico de

$$f_{audible} = f_s - f_{original} = 3985Hz$$

que se corresponde con la altura de la segunda línea roja en el espectrograma de la figura 1.2.

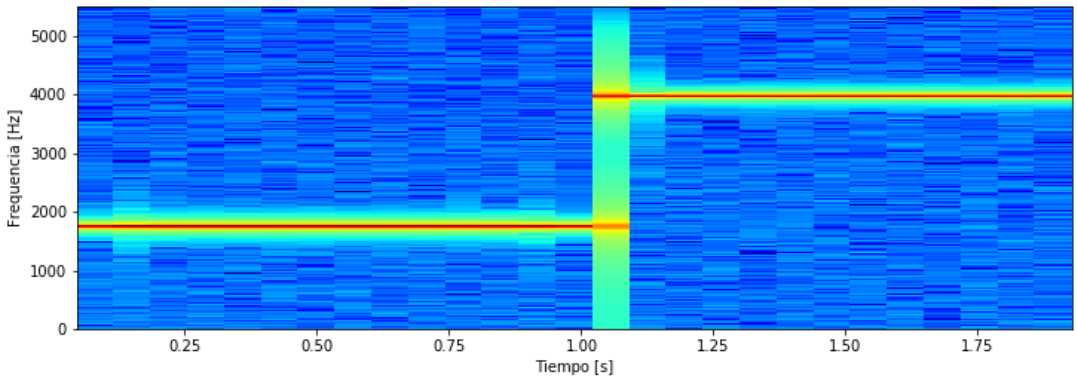


Figura 1.2: Espectrograma de los tonos re-muestreados a 11025 Hz descartando una de cada dos muestras.

Función `decimate`

Al aplicar la función del paquete `scipy.signal`, se puede notar que los cambios antes vistos para el caso de descartar muestras ya no se contemplan, esto se debe a que la función 'decimate' se encarga de aplicar un filtro **antialiasing** antes de realizar el downsampling. Es por eso que en último caso para la frecuencia $f_s = 11025Hz$ el tono original de 7040 Hz desaparece por completo como se puede ver en la figura 1.3, sobreviviendo solo el tono que cumple con las hipóteses de muestreo, es decir, que está por debajo de la frecuencia de Nyquist.

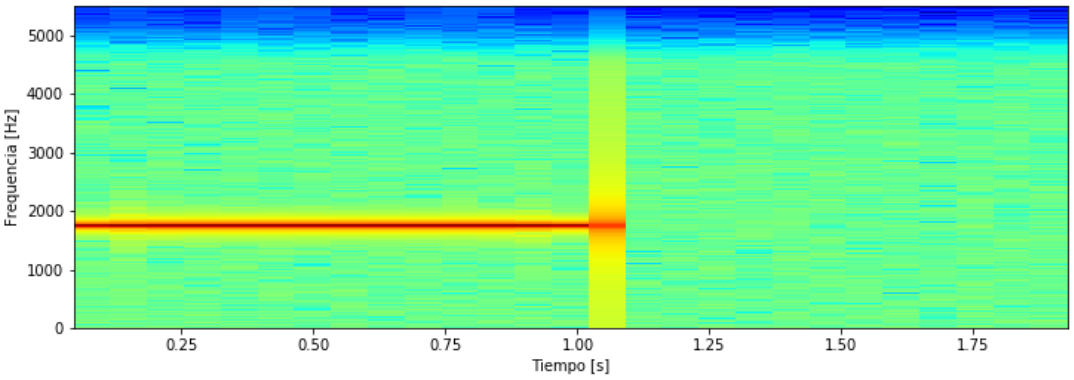


Figura 1.3: Espectrograma de los tonos re-muestreados a 11025 Hz utilizando la función `decimate`

1.1.2. Archivo `chirp.wav`

Todos los procedimientos realizados en la Parte 1 se repiten utilizando un nuevo archivo que contiene un barrido de frecuencia en todo el espectro, es decir, comenzando de una frecuencia baja y aumentando la altura constantemente, como se puede ver en la figura 1.4.

Para el caso de este archivo las observaciones son similares a las correspondientes al archivo anterior, lo más destacable es que al aplicar un submuestreo sin filtro **antialiasing**

se va produciendo una subida y bajada de altura en la señal, lo que deja en evidencia que efectivamente se violan las hipótesis de muestreo y se produce aliasing, este fenómeno se puede ver en la figura 1.5 que corresponde a una $f_s = 11025Hz$ y es donde el efecto es más notorio.

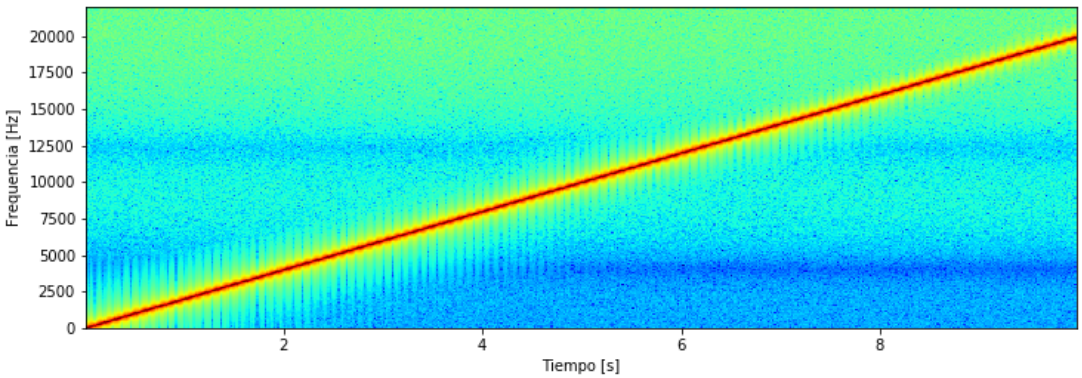


Figura 1.4: Espectrograma del archivo `chirp.wav`

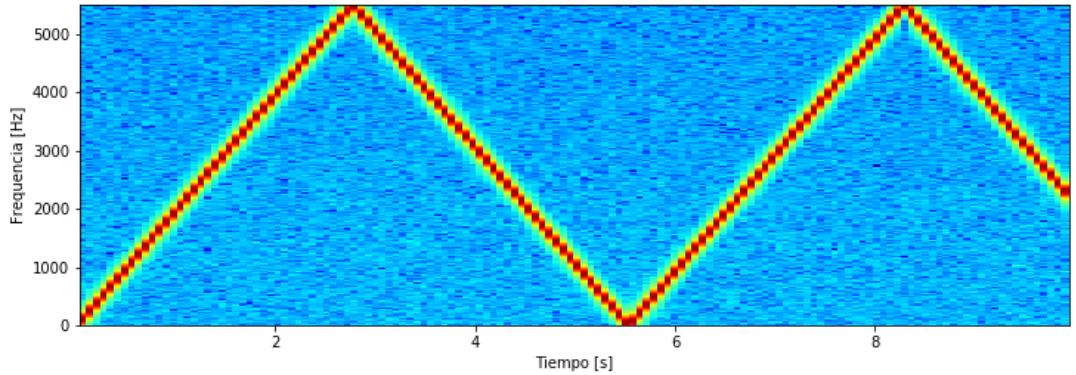


Figura 1.5: Problema de *aliasing* al hacer un submuestreo simplemente descartando muestras, $f_s = 11025Hz$.

Por último y nuevamente, al aplicar la función `decimate` debido al filtro antialiasing que aplica antes de hacer el re-muestreo, se observa que las frecuencias por encima de la frecuencia de Nyquist simplemente desaparecen tal cual ocurrió anteriormente para el archivo de los tonos.

1.2. Parte 2

1.2.1. 1)

Asumiendo las condiciones de la letra del problema, la energía del *error de cuantización* se puede calcular como la integral del error al cuadrado por la densidad de probabilidad. Como lo indica la siguiente expresión

$$E_q = \int_{-\infty}^{+\infty} e^2 p(e) de$$

como el error se asume uniforme en un intervalo de valor Q que representa el paso mínimo de cuantización, la densidad de probabilidad tiene el valor

$$p(e) = \begin{cases} 1/Q & \text{si} \quad e \in [-\frac{Q}{2}, \frac{Q}{2}] \\ 0 & \text{en otro caso} \end{cases}$$

Resolviendo entonces la integral anterior

$$E_q = \frac{1}{Q} \int_{-\frac{Q}{2}}^{+\frac{Q}{2}} e^2 de$$

$$E_q = \frac{Q^2}{12}$$

Por lo que tomando raíz cuadrada obtenemos el error RMS de cuantización

$$E_{qRMS} = \frac{Q}{\sqrt{12}}$$

Por otro lado, si el sistema tiene un largo n de bits de palabra, existen $N = 2^n$ palabras disponibles, de las cuales 2^{n-1} se utilizan para distinguir entre valores positivos y negativos, como habíamos definido un paso mínimo Q de cuantización, se deduce que el pico máximo que puede tomar una señal es de $Pico_{max} = \pm Q2^{n-1}$.

El valor RMS de la señal es entonces

$$S_{RMS} = \frac{Q2^{n-1}}{\sqrt{2}}$$

Por último, el valor S/E o *signal to error* es simplemente el cociente de los valores E_{qRMS} y S_{RMS} elevados al cuadrado.

$$S/E = \frac{(Q2^{n-1})^2}{2} \times \frac{12}{Q^2} = 6 \times 2^{2(n-1)} = \frac{3}{2} 2^{2n} \tag{1.1}$$

En la ecuación anterior se pasó un factor de 2 al exponente, de ahí es que aparece el 3/2 como factor multiplicando al inicio. Para una forma mas intuitiva de analizar esta relación se puede pasar a dB quedando de la siguiente manera

$$S/E_{dB} = 10 \times log\left(\frac{3}{2} 2^{2n}\right)$$

$$S/E_{dB} = 10 \times log\left(\frac{3}{2}\right) + 20n \times log(2)$$

$$S/E_{dB} = 6{,}02n + 1{,}76 \quad dB \tag{1.2}$$

de la ecuación (1.2) concluímos que aumentar un bit en el largo de palabra, repercute en aumentar en $6dB$, o lo que es igual, al doble de la relación S/E

1.2.2. 2)

Para el primer caso, la *pdf* de un dither triangular tiene la siguiente forma

$$pdf_{triang}(x) = \begin{cases} \frac{1}{Q}\left(1 - \frac{x}{Q}\right) & \text{si} \quad x \in [0, Q] \\ \frac{1}{Q}\left(\frac{x}{Q} - 1\right) & \text{si} \quad x \in [-Q, 0] \\ 0 & \text{en otro caso} \end{cases}$$

Integrando en forma partida y aplicando que la *pdf* es simétrica

$$\begin{aligned} E_{triang} &= 2\int_{-Q}^{0} x^2 \frac{1}{Q}\left(1 - \frac{x}{Q}\right) dx \\ &= \frac{2}{Q}\left[\frac{x^3}{3}\Big|_0^Q - \frac{x^4}{4Q}\Big|_0^Q\right] \\ &= \frac{Q^2}{6} \end{aligned}$$

La potencia de ruido de cuantización total al agregar dithering triangular es entonces

$$E_{triang} = \frac{Q^2}{12} + \frac{Q^2}{6} = \frac{Q^2}{4} \tag{1.3}$$

Para el segundo caso, la pdf rectangular aporta exactamente la misma energía que el error de cuantización uniforme, por lo que la energía total pasa a ser

$$E_{unif} = \frac{Q^2}{6} \tag{1.4}$$

Para el último caso, la pdf gaussiana tiene la forma

$$pdf_{gauss}(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Sabiendo que $\sigma^2 = Q^2/4$ y aplicando resultados conocidos sobre la integral

$$\begin{aligned} E &= \int \frac{x^2}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} dx \\ &= \frac{Q^2}{4} \end{aligned}$$

y por consiguiente la potencia sumada es de

$$E_{guss} = \frac{Q^2}{12} + \frac{Q^2}{4} = \frac{Q^2}{3} \tag{1.5}$$

1.2.3. 3)

Para esta parte se creó un tono puro de 500 Hz con amplitud 2 muestreado a 44100 Hz a efectos de trabajar con el, los procedimientos fueron: en primera instancia, cuantizar la señal con pasos de valor $Q = 1$, y posteriormente agregar tres tipos distintos de dither (uniforme, triangular y guassiano) al tono puro y cuantizar estas tres señales nuevas para analizar y comparar los resultados.

En el caso de la señal pura cuantizada, se puede escuchar claramente que se produce una distorsión producto de la baja amplitud de la señal con respecto a los pasos de cuantización. También es posible ver en el espectrograma de la figura 1.6 que aparecen armónicos de dicha señal de manera muy notoria, debido a que la señal deja de tener la forma sinusoidal y pasa a tener más la forma de una onda cuadrada.

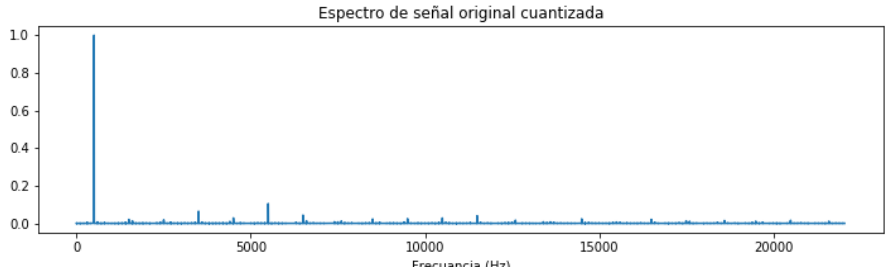


Figura 1.6: Espectro del tono a 500 Hz cuantizado.

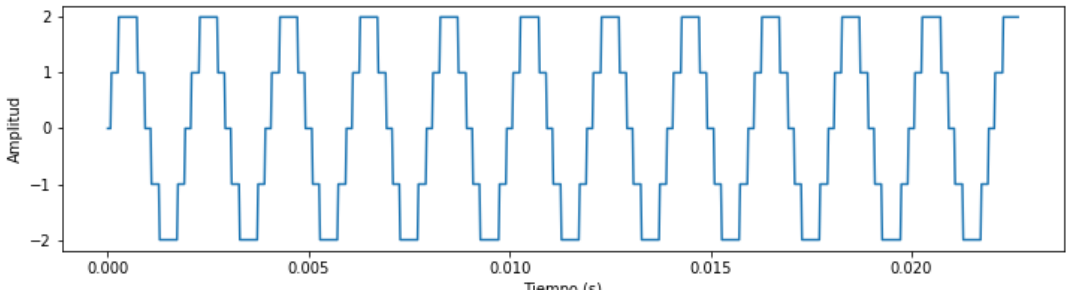


Figura 1.7: Tono a 500 Hz cuantizado. Se muestran solamente 1000 samples a efectos visuales.

Al cuantizar la señal con los distintos tipos de dither agregado desaparece el efecto de distorsión, pero el piso de ruido aumenta considerablemente (que ya se observaba previo al proceso de cuantización), en otras palabras, al aplicar dither es como si pudiéramos obtener niveles de cuantización incluso por debajo del paso mínimo. Otra observación es que gráficamente y preceptivamente se pueden ver las diferentes contribuciones en frecuencia de cada dither, el gaussiano se asemeja mucho a un ruido blanco, mientras que el uniforme da una sensación de ruido más 'equilibrado' en frecuencia.

Por último, en el espectrograma de la señal cuantizada con dither triangular de la figura 1.8, se pueden apreciar dos picos armónicos por encima de la frecuencia fundamental de la sinusoide, que en principio nos lleva a pensar que a pesar de haber aplicado dither,

estos sobreviven en el proceso de cuantización. Esta observación tambien se puede realizar en el espectrograma de la señal, dos líneas más tenues se proyectan por encima de la línea fundamental.

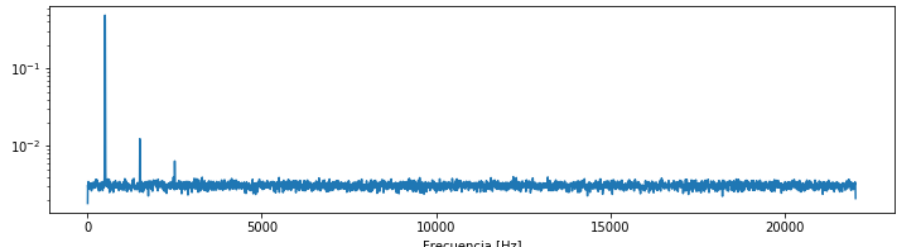


Figura 1.8: Espectro de la señal original con dither triangular cuantizada.

Potencia de las señales

Habiendo hecho los cálculos correspondientes, los valores arrojados para las distintas señales fueron

- Potencia de señal original: 3,01dB

- Potencia de error de cuantizacion dithering uniforme: $-8{,}39dB$
- Potencia de error teórico de cuantizacion dithering uniforme: $-7{,}78dB$

- Potencia de error de cuantizacion dithering gaussiano: $-5{,}67dB$
- Potencia de error teórico de cuantizacion dithering gaussiano: $-4{,}77dB$

- Potencia de error de cuantizacion dithering triangular: $-6{,}65dB$
- Potencia de error teórico de cuantizacion dithering triangular: $-6{,}02dB$

Donde la mayor diferencia se encuentra al haber aplicado dither de tipo gaussiano, en el resto de los casos los valores obtenidos se asemejan a los valores esperados por lo que no hay mucho que profundizar al respecto.

2. Problema 2 - Características en tiempo corto de una señal de audio

2.1. Parte 1

2.1.1. 1)

Para la ventana w_a, todos los términos tales que $[n-m] \geq 0$ son los que aportan a la sumatoria, por lo que se debe cumplir que $m \leq n$, aplicando estas condiciones y la forma

explicita de la ventana tenemos

$$E_n = \sum_{m=-\infty}^{\infty} x^2[m] w_a[n-m] = \sum_{m=-\infty}^{n} x^2[m] a^{n-m} \tag{2.1}$$

observar, que de igual manera,

$$E_{n-1} = \sum_{m=-\infty}^{\infty} x^2[m] w_a[(n-1)-m] = \sum_{m=-\infty}^{n-1} x^2[m] a^{(n-1)-m}$$

de donde extrayendo el último sumando al evaluar $m = n$ en el término de la derecha en (2.1) y sacando un factor de a hacia afuera de la sumatoria obtenemos

$$E_n = \sum_{m=-\infty}^{n} x^2[m] a^{n-m} = x^2[n] + a \sum_{m=-\infty}^{n-1} x^2[m] a^{(n-1)-m}$$

Que es exactamente igual que

$$E_n = x^2[n] + a E_{n-1} \tag{2.2}$$

Otra manera de verlo, es pensar que al agregar otra muestra, debería multiplicar todos los sumandos anteriores por el valor de a correspondiente a la vetnana w_a y luego sumar el valor de la muestra $x^2[n]$ que se desea agregar, ya que al evaluar el límite superior en la ventana resulta $w_a = 1$.

Para el caso de la magnitud en tiempo corto el razonamiento es análogo, se extrae el último sumando de la sumatoria, y se saca un factor a de la sumatoria restante, quedando

$$M_n = \sum_{m=-\infty}^{\infty} |x[m]| w_a[n-m] = \sum_{m=-\infty}^{n} |x[m]| a^{n-m}$$

$$M_n = |x[n]| + \sum_{m=-\infty}^{n-1} |x[m]| a^{n-m} = |x[n]| + a \sum_{m=-\infty}^{n-1} |x[m]| a^{(n-1)-m}$$

Concluyendo que

$$M_n = |x[n]| + a M_{n-1} \tag{2.3}$$

Energía y Magnitud de tiempo corto para identificar palabras.

Para realizar el cálculo de la energía y la magnitud de tiempo corto en el archivo `voice.wav` se utilizó una ventana rectangular de $40msec$. Arrojando como resultado las medidas graficadas en las figuras 2.1 y 2.2. A priori, se podría decir que para el caso de la energía se podrían identificar a simple vista donde están ubicadas las distintas palabras en el audio, no obstante existen regiones donde la energía es grande pero no se corresponde con palabra alguna, sino a distintos sonidos ya sean producto de ruido u otros factores, que no forman parte del speech. Un detalle importante es que los valores de la energía para los fonemas tonales es bastante superior que la correspondiente a los fonemas fricativos o sordos, y esto se debe que al computarse la energía se utiliza el cuadrado de la señal, y es de gran ayuda para distinguir las transiciones entre los fonemas tonales y los sordos.

En el caso de la magnitud, las diferencias entre los sonidos sordos y tonales ya no es tan amplia, ya que se implementa el módulo de la señal para hacer el cálculo y no el cuadrado como en el caso anterior. Aunque en principio esta medida arroja un resultado un poco más 'ruidoso' a simple vista en términos de que da un espectro de señal más parejo, aún es posible teniendo los cuidados pertinentes, distinguir donde se producen las palabras sobre todo en la primera mitad del archivo.

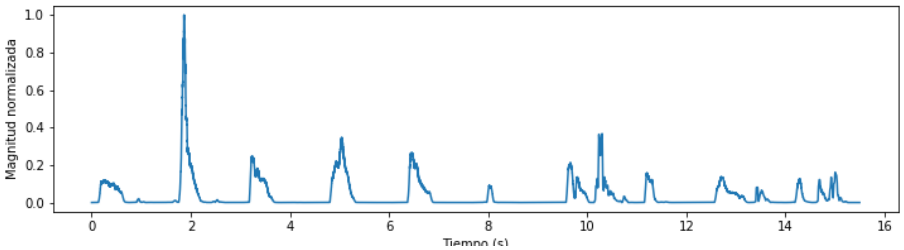


Figura 2.1: Energía de tiempo corto con ventana de 40msec para `voice.wav`

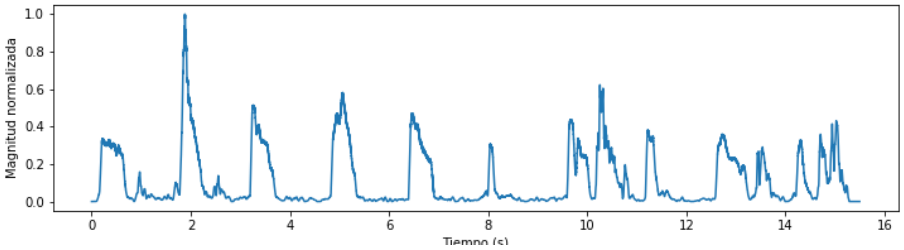


Figura 2.2: Magnitud de tiempo corto con ventana de 40msec para `voice.wav`

2.2. Parte 2

2.2.1. 1)

La tasa de cruce por ceros se define como

$$Z_n = \sum_{m=-\infty}^{\infty} |\text{sign}(x[m]) - \text{sign}(x[m-1])| w_n[n-m] \tag{2.4}$$

$$\text{donde} \quad \text{sign}(x[n]) = \begin{cases} 1, & x[n] \geq 0 \\ -1, & x[n] < 0 \end{cases} \quad \text{y} \quad w_n = \begin{cases} \frac{1}{2N}, & 0 \leq n \leq N-1 \\ 0, & \text{en otro caso.} \end{cases}$$

A partir de la ecuación (2.4) y aplicando el enventanado, podemos expresar la sumatoria como

$$Z_n = \frac{1}{2N} \sum_{m=n-N-1}^{n} |\text{sign}(x[m]) - \text{sign}(x[m-1])| \tag{2.5}$$

Antes de seguir con el desarrollo es conveniente observar que

$$Z_{n-1} = \frac{1}{2N} \sum_{m=n-N}^{n-1} |\text{sign}(x[m]) - \text{sign}(x[m-1])| \tag{2.6}$$

Desarrollando (2.5), como primer paso desacoplamos el último término de la sumatoria, es decir el término correspondiente a $m = n$, obteniendo

$$Z_n = \frac{1}{2N} |\text{sign}(x[n]) - \text{sign}(x[n-1])| + \frac{1}{2N} \sum_{m=n-N-1}^{n-1} |\text{sign}(x[m]) - \text{sign}(x[m-1])| \tag{2.7}$$

Se puede ver que la sumatoria del lado derecho de la ecuación (2.7) contiene un término menos que la ecuación (2.6), si agregamos ese término en la sumatoria y lo restamos fuera de ella

$$Z_n = \frac{1}{2N} \{ |\text{sign}(x[n]) - \text{sign}(x[n-1])| - |\text{sign}(x[n-N]) - \text{sign}(x[n-N-1])| \} + \frac{1}{2N} \sum_{m=n-N}^{n-1} |\text{sign}(x[m]) - \text{sign}(x[m-1])| \tag{2.8}$$

donde el segundo término a la derecha de la ecuación (2.8) es exactamente igual que Z_{n-1} dado por (2.6). Sustituyendo esto queda como resultado final la ecuación en recurrencia

$$Z_n = Z_{n-1} + \frac{1}{2N}\{|\text{sign}(x[n]) - \text{sign}(x[n-1])| - |\text{sign}(x[n-N]) - \text{sign}(x[n-N-1])|\} \quad (2.9)$$

Tasa de cruces por cero para el archivo `voice.wav` utilizando ventana de 10 msec

Al aplicar el cálculo al archivo con una ventana de tamaño $N = f_s \times \tau$ con $\tau = 10msec$ obtenemos lo visto en la figura 2.3. Visualmente a grandes rasgos uno podría distinguir donde se encuentran las palabras en el gráfico, pero lo cierto es que esta medida por si sola no alcanza para poder distinguir comienzos y finales de palabras.

Otro posible intento es agrandar la ventana temporalmente, utilizar por ejemplo 80 msec, si bien esto hace que la gráfica sea más suave, es decir las variaciones temporales son mas lentas (ver figura 2.4), aún así distinguir las palabras solamente a partir de esta medida sigue siendo prácticamente inviable.

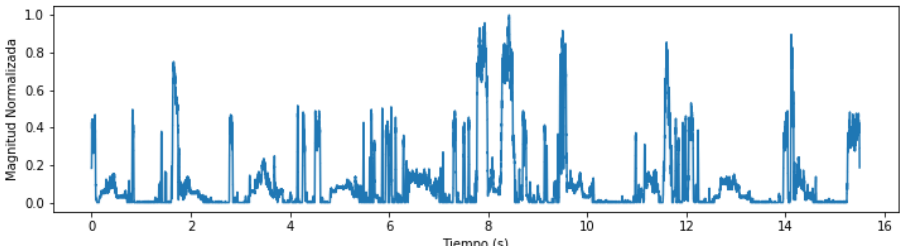


Figura 2.3: Tasa de cruces por cero para el archivo `voice.wav` con ventana de 10msec.

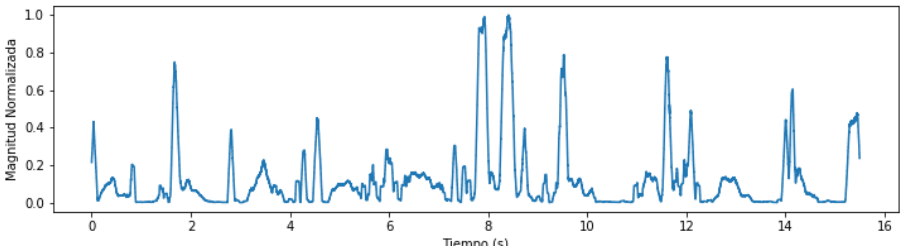


Figura 2.4: Tasa de cruces por cero para el archivo `voice.wav` con ventana de 80msec.

2.2.2. Detección de palabras utilizando la energía y tasa de cruces por cero.

Para abordar este problema, en principio se tuvo en cuenta solamente el reconocimiento de fonemas tonales, es decir, aquellos en los que la energía es alta y la tasa de cruces por cero es baja. Seteando los umbrales correspondientes en el algoritmo se recogieron resultados no tan insatisfactorios. Todos los fonemas tonales claramente marcados fueron detectados, pero trajo el problema de no reconocer exactamente el inicio y final en algunos

casos como un oído los detectaría fruto del análisis más complejo que naturalmente uno hace (si le antecede o precede un silencio, o estar concatenado a un fonema sordo, distinguir naturalmente ruido de no ruido, etc).

Al ser una función bastante genérica y simple, depende tambíen de la calidad del archivo a analizar, por ejemplo en el archivo `voice.wav` se obtuvieron resultados significativamente mejores que en los archvos `fox.wav` y `voice2.wav`, sobre todo ante este último.

En un intento de afinar el algoritmo, se filtró la señal de nuevo pero en este caso apelando a detectar 'ruido' o sonidos de bajas frecuencias que no corresponden al rango vocal humano. Si bien no arrojó resultados extraordinarios, en algunos casos sí funcionó y quitó segmentos que antes reconocía como palabras pero en realidad no lo eran, a pesar de que quedaron algunos segmentos "basura". Un caso marcado se puede ver en la figura 2.5 para el archivo `voice.wav`.

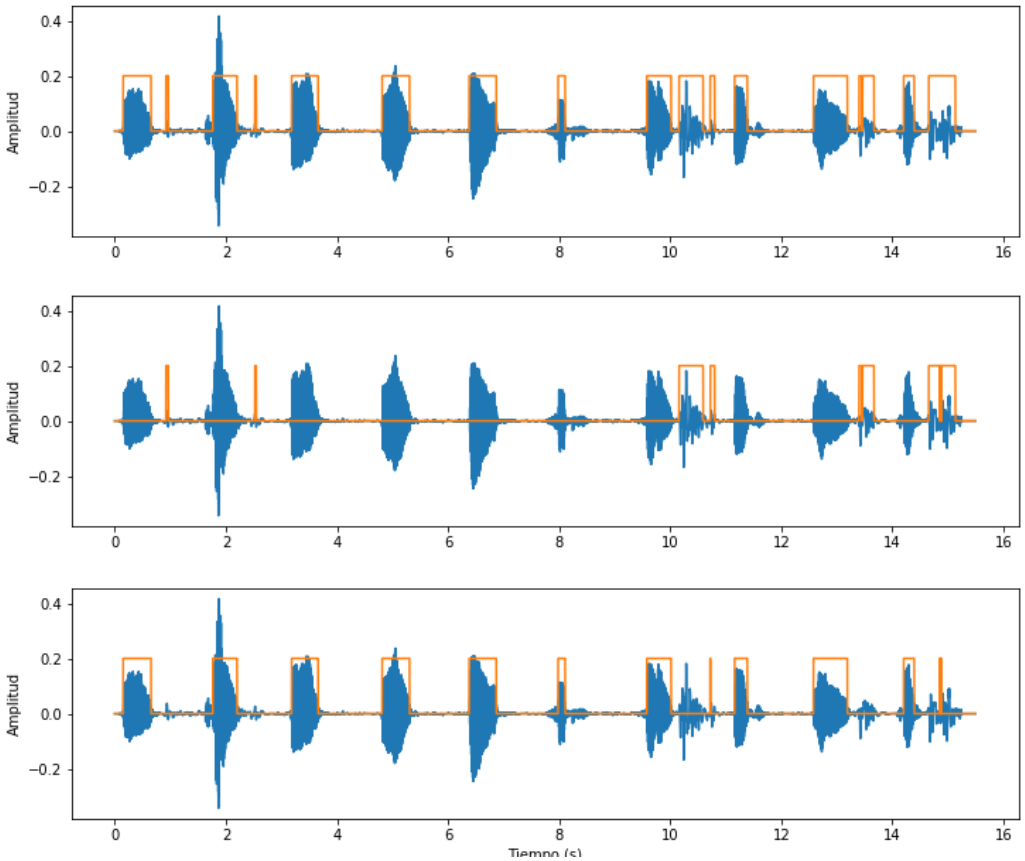


Figura 2.5: Primero detección normal. Segundo detección de 'basura'. Tercero resta de ambas.

Otro paso posterior fue intentar detectar fonemas sordos, aquellos que tengan una alta tasa de cruces por cero pero poca energía. En este aspecto los resultados no fueron tan buenos como en la detección de sonidos tonales. Se destaca el resultado en el archivo `fox.wave`, ya que contiene bien pronunciados fonemas como la **'f'**, **'x'** y **'z'**. Si bien también hay porcentaje de detección 'basura', los sonidos sordos correspondientes a los fonemas anteriormente mencionados se detectaron por completo, como se puede ver en la

figura 2.6.

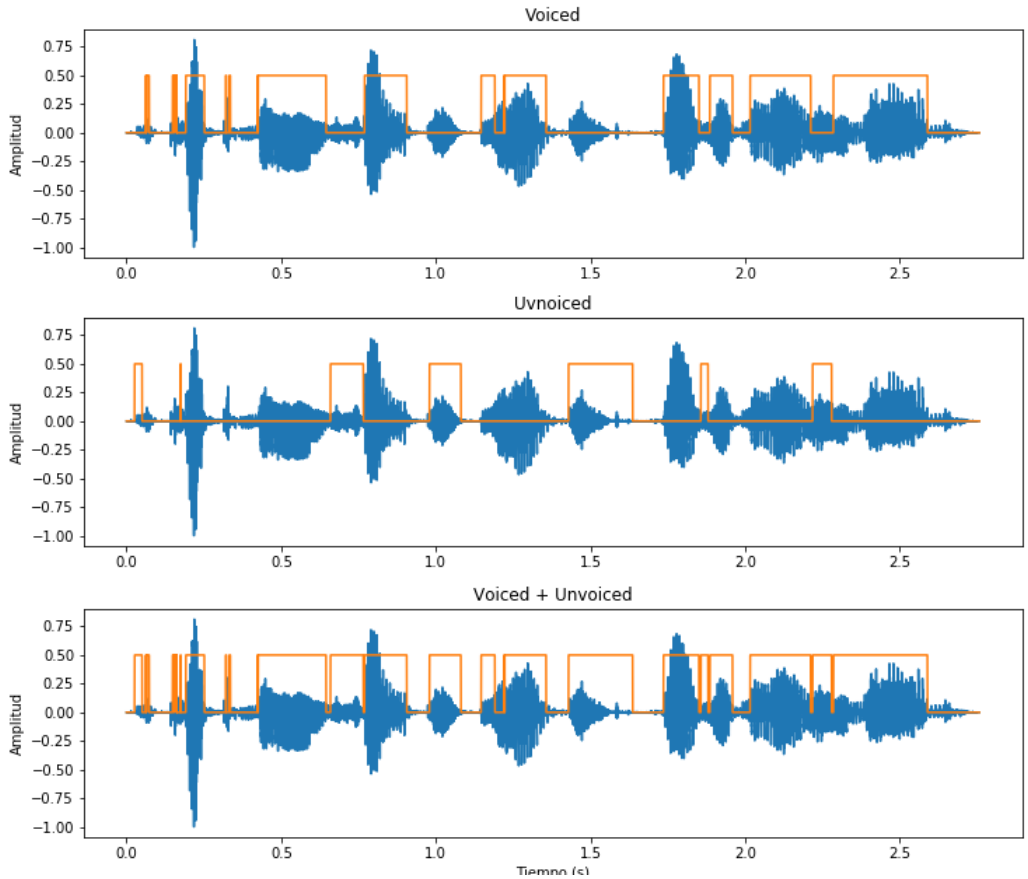


Figura 2.6: Detección de sonidos tonales y sordos para el archivo `fox.wav`

Por último se muestra el caso del archivo `voce2.wav` solamente con la detección de sonidos tonales en la figura 2.7, ya que los resultados obtenidos al aplicar la detección de fonemas sordos no fueron satisfactorios.

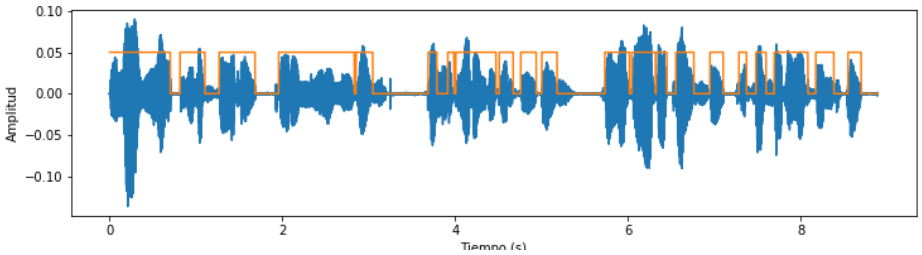


Figura 2.7: Detección de sonidos tonales para el archivo `voice2.wav`

Las principal limitación para abordar este problema es la calidad del archivo a analizar, quizás para una una grabación de voz de muy buena calidad, contemplar simplemente los niveles de energía y tasa de cruces por cero sea suficiente para obtener muy buenos reultados a la hora de detectar las palabras, pero cuando la calidad no es tan superior

algunas situaciones pueden verse camufladas por esto, por ejemplo que la energía del piso de ruido sea semejante a la de un sonido sordo, lo cual tentaría al algoritmo a confundirse.

Otro aspecto a tener en cuenta es que se debería contemplar también todos los niveles de la comunicación oral (la semántica, sintaxis, articulación, etc) ya que de esta manera aún reconociendo ciertos sonidos como palabras o fonemas, a partir de este otro análisis podrían filtrarse y discutirse para lograr un resultado mas refinado.

3. Problema 3 - Cálculo y aplicación de la función de autocorrelación

3.1. Parte 1 - Propiedades de la autocorrelación

3.1.1. 1)

Se define la función de autocorrelación en tiempo corto como

$$R_n[k] = \sum_{m=-\infty}^{\infty} x[m]w[n-m]x[m+k]w[n-k-m] \tag{3.1}$$

Para probar que esta función es par, es decir $R_n[k] = R_n[-k]$ basta sustituir en la ecuación (3.1) el valor $m' = m + k$, o lo que es lo mismo $m = m' - k$, operando queda

$$R_n[k] = \sum_{m=-\infty}^{\infty} x[m]w[n-m]x[m+k]w[n-k-m] = \sum_{m'=-\infty}^{\infty} x[m'-k]w[n-m'+k]x[m']w[n-m'] = R_n[-k] \tag{3.2}$$

que no es otra cosa que la autocorrelación evaluada en $-k$.

3.1.2. 2)

Aplicando el resultado anterior y haciendo algunas manipulaciones

$$R_n[k] = R_n[-k] = \sum_{m=-\infty}^{\infty} x[m]x[m-k]w[n-m]w[n-m+k] \tag{3.3}$$

$$= \sum_{m=-\infty}^{\infty} x[m]x[m-k]h_k[n-m] \tag{3.4}$$

donde $h_k[n] = w[n]w[n+k]$

3.2. Parte 2 - Estimación de la frecuencia fundamental

Para esta parte se utilizó la función de autocorrelación para detectar la frecuencia fundamental en la señal de aduio del archivo `LP-mem-6-a.wav`, que corresponde a una voz femenina cantando *a cappella*, muestreada a 24.000 Hz.

Los pasos a seguir para el algoritmo fueron los siguientes:

1. Se toman intervalos de 30 ms de largo, cada 10 ms (i.e. 20 ms de superposición).
2. Para cada intervalo:
    1. Se calculan los primeros 250 valores de la función de autocorrelación.
    2. Se busca el índice k_f del primer máximo local que supere el 60 % del valor en cero. Si existe, hay un componente periódico lo suficientemente notorio, cuya frecuencia es $f = \frac{f_s}{k_f}$
    3. Si no se encuentra ningún máximo local bajo esa condición, se asigna $f = 0$, considerándose que no hay un componente periódico relevante en ese intervalo de tiempo.
3. Se obtiene un vector con valores de frecuencia cada 10 ms comenzando en $t = 0{,}015s$

Una vez implementado el algoritmo, se corrió sobre el archivo utilizando los parámetros descritos en los pasos anteriormente mencionados y una ventana de tipo rectangular. Arrojando como resultado lo que se puede ver en la figura 3.1.

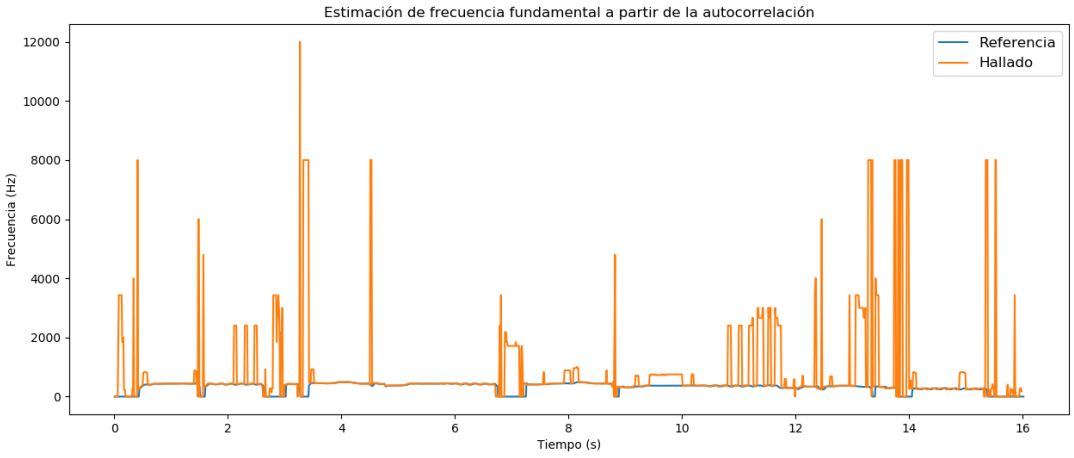


Figura 3.1: Estmiación de frecuencia fundamental para archivo `LP-mem-6-a.wav` inicial.

Los resultados para este caso no fueron para nada alentadores, ya que hay picos de frecuencias que llegan a estar a una altura de $12{,}000Hz$, que es esencialmente la frecuencia de Nyquist para este caso. Considerando que se trata de una voz humana, y que difícilmente esa frecuencia corresponda a algún tipo de ruido en la grabación, se dedujo que claramente este fenómeno corresponde a un error del algoritmo. También se aprecian frecuencias del orden de los $2500Hz$ a $8000Hz$.

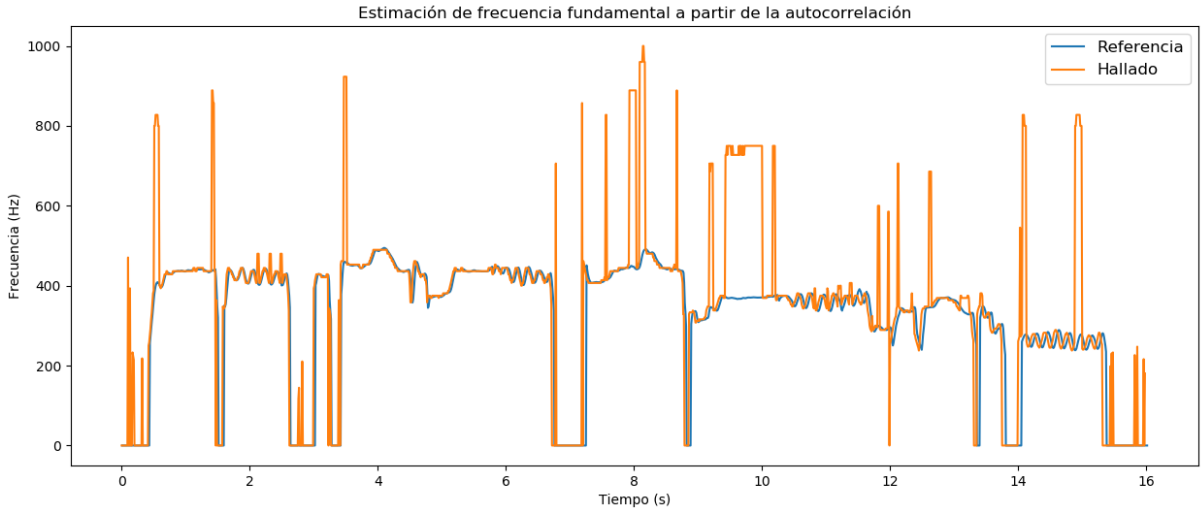


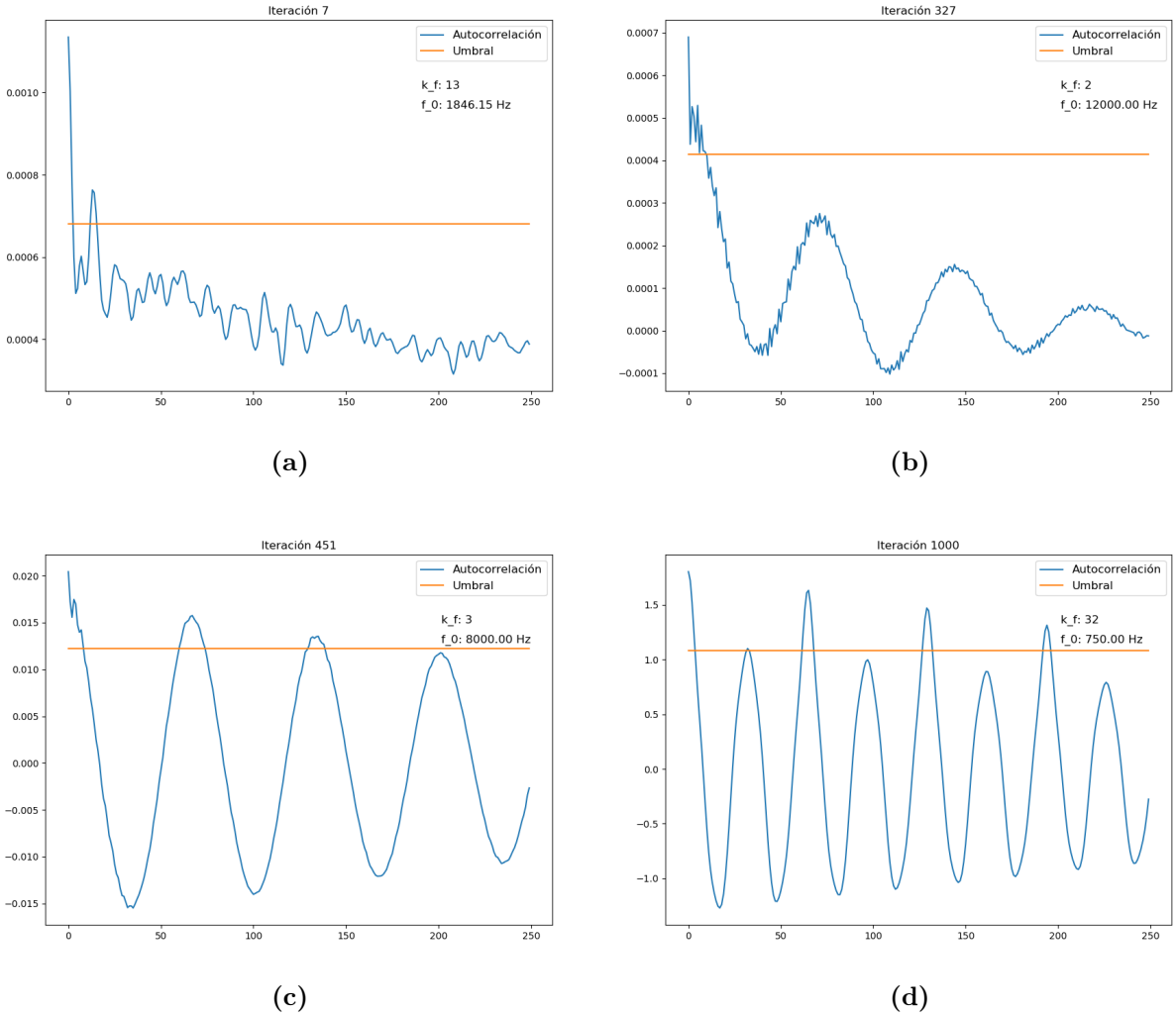
Figura 3.2: Estmiación de frecuencia fundamental para archivo `LP-mem-6-a.wav` con mayor entorno de máximo local.

Cabe destacar que para hallar el máximo local se contempló solamente que la amplitud del índice inmediato anterior y el índice inmediato posterior fueran menores que el máximo, siendo la magnitud del máximo mayor al umbral como era requisito. Es decir:

$$\begin{cases} \texttt{autocorrelacion[kf-1]} < \texttt{autocorrelacion[kf]} \\ \texttt{autocorrelacion[kf]} > \texttt{Umbral} \\ \texttt{autocorrelacion[kf+1]} \leq \texttt{autocorrelacion[kf]} \end{cases}$$

Para comprender mejor la razón de estos picos de frecuencia se buscaron los instantes donde ocurrieron y se estudió la autocorrelación más detenidamente. Observando para algunos casos elegidos aleatoriamente, se puede ver en la figura 3.3 la representación de cuatro ventanas de autocorrelación de largo 250 para cuantro instantes temporales distintos. La figura **(a)** corresponde a una ventana situada al inicio del audio (recordar que el número de iteración corresponde a un desplazamiento de 10ms con respecto a la posición inicial) donde se esperaba encontrar un valor de $f_0 = 0$. La figura **(b)** corresponde al instante donde se alcanza el pico de $2500Hz$, se puede ver que la razón de esto es que existe una perturbación de muy alta frecuencia enseguida que arranca la ventana, por lo que el algoritmo detecta que el primer pico de dicha perturbación cae dentro de las hipótesis de máximo local. Con la figura **(c)** ocurre algo parecido que con el caso **(b)**, con la salvedad que mientras que en el caso anterior no existía componente periódico notorio, aquí si lo hay pero no se detecta correctamente fruto nuevamente de una perturbación de alta frecuencia al inicio del enventanado. Por último en la figura **(d)** se puede intuir visualmente que el pico máximo local que correspondería a la verdadera frecuencia fundamental es el que se genera justo después del pico detectado por el algoritmo.

Para algunos de estos casos, la solución más eficiente fue la de extender el entorno definido para el máximo local, aumentando la cantidad de valores previos que deben estar por debajo del máximo para que este se identifique como tal. Es decir nuevamente:



(a) **(b)**

(c) **(d)**

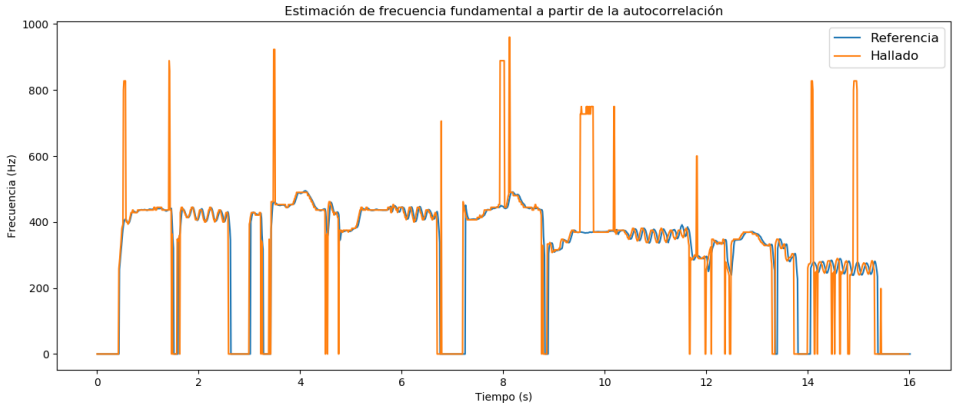
Figura 3.3: Cuatro casos distintos de ventanas de autocorrelación para la señal de audio.

$$\begin{cases} \texttt{autocorrelacion[kf-n:kf]} < \texttt{autocorrelacion[kf]} \\ \texttt{autocorrelacion[kf]} > \texttt{Umbral} \\ \texttt{autocorrelacion[kf+1]} \leq \texttt{autocorrelacion[kf]} \end{cases}$$

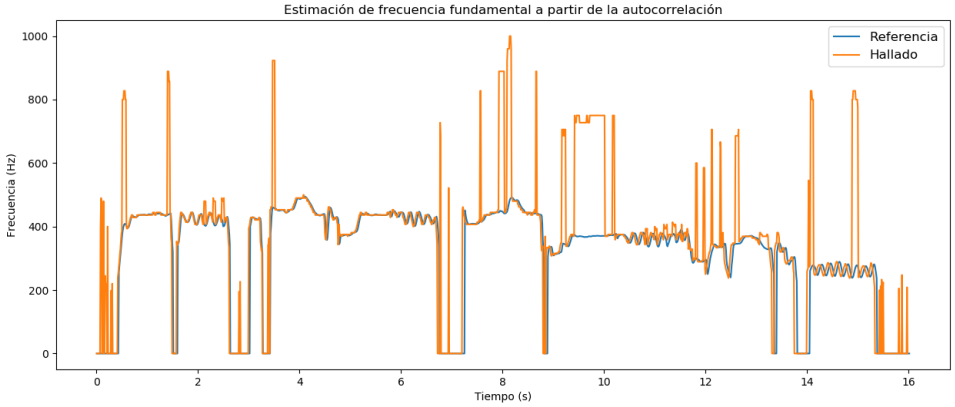
Computando este nuevo algoritmo el resultado arrojado fue el que se puede ver en la figura 3.2, que si bien el resultado es notoriamente superior al inicial, aún se observan perturbaciones en la detección de la frecuencia fundamental. Se probaron otras configuraciones como utilizar una ventana de Hamming en vez de una ventana rectangular, o aumentar levemente el umbral para deshacerse de posibles casos como el que ocurre en la figura 3.3(d). Algunos de estos resultados se pueden observar en la figura 3.4.

Considero que este algoritmo a pesar de no ser tan complejo da buenos resultados, o al menos se asemeja a los resultados que uno podría apelar a llegar. Seguramente refinando los casos bordes y agregando alguna otra herramienta que permita ser más preciso a la

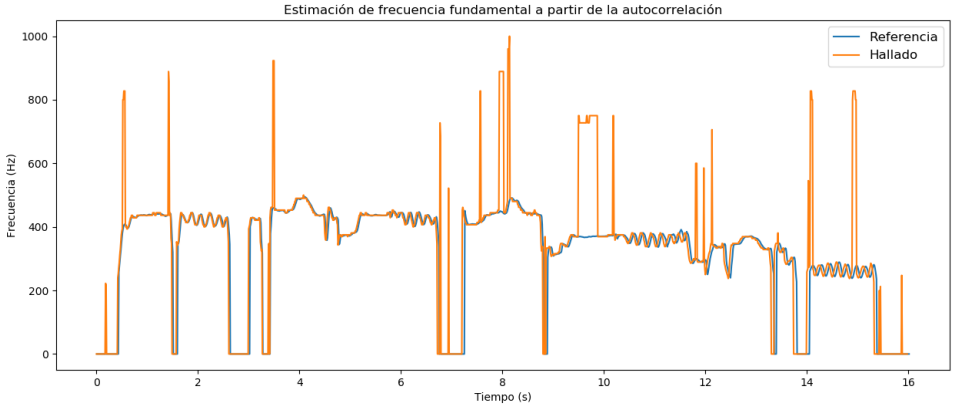
hora de detectar posibles picos que signifiquen una periodicidad marcada, por ejemplo poder reducir la cantidad de casos que se muestran en la figura 3.3, ya implicaría un rendimiento notoriamente mejor.



(a)



(b)



(c)

Figura 3.4: **(a)** Umbral seteado al 70 % del valor en cero. **(b)** Ventana tipo Hamming con umbral al 60 % del valor en cero. **(c)** Ventana tipo Hamming con umbral al 70 % del valor en cero.