

**ALGORITMO DE DECONVOLUCIÓN DISCRETA  
OPTIMIZADO PARA APLICACIONES EN ACÚSTICA:  
ADOA**

**ANTONIO SANDOVAL MARTÍNEZ**

**UNIVERSIDAD DE SAN BUENAVENTURA  
FACULTAD DE INGENIERÍA  
INGENIERÍA DE SONIDO  
BOGOTÁ D.C. - COLOMBIA**

**2006**

**ALGORITMO DE DECONVOLUCIÓN DISCRETA  
OPTIMIZADO PARA APLICACIONES EN ACÚSTICA:  
ADOA**

**ANTONIO SANDOVAL MARTÍNEZ**

**UNIVERSIDAD DE SAN BUENAVENTURA  
FACULTAD DE INGENIERÍA  
INGENIERÍA DE SONIDO  
BOGOTÁ D.C. - COLOMBIA**

**2006**

# TABLA DE CONTENIDO

<b>1</b>	<b>PLANTEAMIENTO DEL PROBLEMA.....</b>	<b>6</b>
1.1	ANTECEDENTES.....	6
1.2	DESCRIPCIÓN Y FORMULACIÓN DEL PROBLEMA .....	6
1.3	JUSTIFICACIÓN .....	7
1.4	OBJETIVOS DE LA INVESTIGACIÓN .....	8
1.5	ALCANCES Y LIMITACIONES DEL PROYECTO.....	8
<b>2</b>	<b>MARCO DE REFERENCIA .....</b>	<b>10</b>
2.1	MARCO CONCEPTUAL .....	10
2.2	MARCO TEÓRICO .....	51
<b>3</b>	<b>METODOLOGÍA .....</b>	<b>57</b>
3.1	ENFOQUE DE LA INVESTIGACIÓN .....	57
3.2	LÍNEA DE INVESTIGACIÓN DE USB / SUB-LÍNEA DE FACULTAD / CAMPO TEMÁTICO DEL PROGRAMA LINEA DE INVESTIGACION.....	57
3.3	TÉCNICAS DE RECOLECCIÓN DE INFORMACIÓN.....	58
3.4	HIPÓTESIS.....	58
3.5	VARIABLES .....	58
<b>4</b>	<b>DESARROLLO INGENIERÍL.....</b>	<b>60</b>
4.1	METODOLOGÍA DE DESARROLLO DEL ALGORITMO .....	60
4.2	METODOLOGÍA DE MEDICIONES.....	82
<b>5</b>	<b>PRESENTACIÓN Y ANÁLISIS DE RESULTADOS.....</b>	<b>86</b>
5.1	INTRODUCCIÓN .....	86
5.2	SISTEMAS ELÉCTRICOS .....	86
5.3	SISTEMAS ACÚSTICOS .....	89
5.4	ANÁLISIS DE RESULTADOS.....	93
<b>6</b>	<b>CONCLUSIONES.....</b>	<b>94</b>
<b>7</b>	<b>RECOMENDACIONES .....</b>	<b>96</b>
<b>8</b>	<b>BIBLIOGRAFÍA.....</b>	<b>98</b>
<b>9</b>	<b>ANEXOS .....</b>	<b>99</b>
9.1	GRAFICOS.M.....	99

# TABLA DE ILUSTRACIONES

FIGURA 1 ILUSTRACIÓN DEL OÍDO.....	10
FIGURA 2 HUESECILLOS DEL OÍDO MEDIO.....	12
FIGURA 3 CONEXIÓN DEL OÍDO MEDIO CON EL OÍDO INTERNO .....	14
FIGURA 4 LABERINTO MEMBRANOSO .....	14
FIGURA 5 CURVAS DE IGUAL SONORIDAD .....	18
FIGURA 6 CURVAS DE PONDERACIÓN DE IGUAL SONORIDAD. ....	19
FIGURA 7 REFLEXIÓN, ABSORCIÓN Y TRANSMISIÓN DE UNA ONDA SONORA .....	22
FIGURA 8 SONIDO DIRECTO. ....	24
FIGURA 9 REFLEXIONES TEMPRANAS. ....	25
FIGURA 10 REVERBERACIÓN.....	26
FIGURA 11 DISTANCIA CRÍTICA .....	27
FIGURA 12 REVERBERACIÓN.....	28
FIGURA 13 EFECTOS DE REFLEXIONES EN LA PERCEPCIÓN.....	29
FIGURA 14 EFECTOS DE LA REVERBERACIÓN EN LA INTELIGIBILIDAD DE LA PALABRA.....	31
FIGURA 15 FLUCTUACIÓN DEL DÓLAR.....	33
FIGURA 16 DIFERENCIA ENTRE TIEMPO CONTINUO Y TIEMPO DISCRETO. ....	34
FIGURA 17 INVERTIBILIDAD DE UN SISTEMA .....	37
FIGURA 18 AMPLIFICADOR RETROALIMENTADO.....	38
FIGURA 19 AMPLIFICADOR E INVARIANCIA EN EL TIEMPO.....	39
FIGURA 20 DESCOMPOSICIÓN DE UNA SEÑAL DISCRETA EN IMPULSOS UNITARIOS.....	43
FIGURA 21 ANÁLISIS DE FOURIER PARA UNA SEÑAL DISCRETA.....	46
FIGURA 22 RESOLUCIÓN ESPECTRAL.....	48
FIGURA 23 TAMAÑO DE LA DFT VS. NÚMERO DE OPERACIONES ARITMÉTICAS.....	49
FIGURA 24 EFICIENCIA DEL FFT VS. TAMAÑO DE LA SECUENCIA.....	50
FIGURA 25 SISTEMA DE RETRASOS .....	54
FIGURA 26 RESPUESTA AL IMPULSO DEL SISTEMA DE RETRASOS. ....	86
FIGURA 27 SEÑAL ALTERADA POR EL SISTEMA DE RETRASOS. ....	87
FIGURA 28 AUTOCORRELACIÓN SEÑAL RETRASADA (SE). ....	88
FIGURA 29 COMPARACIÓN RESULTADOS ADOA Y SEÑAL ORIGINAL (SE) .....	89
FIGURA 30 RESPUESTA AL IMPULSO DEL SISTEMA ACÚSTICO EVALUADO (SA) .....	90
FIGURA 31 RESULTADOS ADOA (SA) .....	90
FIGURA 32 COMPARACIÓN ESPECTRAL RESULTADOS ADOA Y SEÑAL ORIGINAL (SA) .....	91
FIGURA 33 COMPARACIÓN RESULTADOS ADOA Y SEÑAL ORIGINAL (SA).....	92
FIGURA 34 PANTALLA PRINCIPAL DEL ADOA. ....	80
FIGURA 35 ESQUEMA DE MEDICIONES ACÚSTICAS.....	84
FIGURA 36 ESQUEMA DE MEDICIONES DE SISTEMAS ELÉCTRICOS.....	85

# **ALGORITMO DE DECONVOLUCIÓN DISCRETA OPTIMIZADO PARA APLICACIONES EN ACÚSTICA: ADOA**

## **INTRODUCCIÓN**

La deconvolución ha sido muy utilizada en el procesamiento digital de imágenes desde hace algunos años para corregir imperfecciones en estas, como fotos con efecto “blur” o borroso. En la acústica ha sido utilizada principalmente para obtener la respuesta al impulso de un sistema acústico a partir de una señal de entrada conocida, y una señal de salida que también se conoce. El algoritmo a desarrollar en este proyecto, pretende determinar la señal de entrada del sistema, a partir de la respuesta al impulso de este y de su salida, es decir, hallar una copia anecoica de la señal de salida del sistema original.

La señal de entrada, dado el caso de un recinto, será convolucionada naturalmente con la respuesta al impulso de dicho sistema. Debido a reflexiones características del sitio, la señal de salida se modificará con respecto a la de entrada. La deconvolución que se aplicará en este proyecto pretende llevar esta señal alterada por las propiedades del recinto a la señal original.

# **1 PLANTEAMIENTO DEL PROBLEMA**

## ***1.1 ANTECEDENTES***

Los antecedes de este proyecto se remontan a 1940 con la construcción de la primera cámara anecoica, diseñada por Murray Hill. La cámara absorbe un 99.995% de las ondas incidentes sobre sus superficies con frecuencias superiores a 200Hz. Pocos años después de esto, se comenzaron a utilizar salas de tiempos de reverberación variables en los estudios de grabación, para obtener diferentes ambientes sonoros.

Hoy en día, la deconvolución es aplicada principalmente al procesamiento digital de imágenes, aunque en acústica también se ha utilizado para obtener la respuesta al impulso de un sistema mediante una señal de entrada y de salida conocidas.

Algunos programas comerciales como el Dirac, actualmente propiedad de Brüel & Kjær y el EASERA, propiedad de SDA, utilizan este último tipo de deconvolución, es decir, hallan la respuesta al impulso del sistema, mediante su entrada y su salida.

## ***1.2 DESCRIPCIÓN Y FORMULACIÓN DEL PROBLEMA***

De forma concreta, el problema es la convolución propia de los sistemas acústicos que afecta la señal de entrada a este, alterándola según las características propias de este. Esta “alteración” pasa a ser la señal de salida. El problema en el caso de un recinto, serían las reflexiones características del sitio y otros fenómenos que modifican la señal de entrada. Dependiendo del comportamiento del sitio o del objetivo del estudio, puede que esta modificación no sea deseada, y es precisamente esto lo que se pretende eliminar.

En general, el problema se manifiesta debido a las características de respuesta del oído humano, ya que este requiere de condiciones específicas de tiempos de reverberación según el uso del recinto.

También se presenta el problema cuando se requiere estudiar un fenómeno acústico sin tener en cuenta ningún tipo de reflexiones, como puede ser el factor de directividad de una fuente o la percepción subjetiva del oído.

### ***1.3 JUSTIFICACIÓN***

Las cámaras anecoicas son recintos muy costosos de construir, además de ocupar grandes volúmenes (generalmente alrededor de 100 metros cúbicos), por lo que no es fácil contar con dichas instalaciones necesarias para el estudio del comportamiento del sonido.

Lo que se pretende lograr con este algoritmo de deconvolución es eliminar la necesidad (en la mayoría de los casos) de usar una cámara anecoica. En la acústica, esto ayudará a estudiar diferentes fuentes emisoras de sonido, el comportamiento oído humano, y en general cualquier estudio o aplicación que requiera analizar el sonido en campo abierto, sin necesidad de que las mediciones sean hechas al aire libre (que generalmente nunca se hace en condiciones anecoicas). En el campo de la grabación y producción se elimina la necesidad de un recinto de sonoridad apropiada, debido a que la señal anecoica puede ser procesada con algoritmos de convolución que dependen de la respuesta al impulso de un sistema ajeno, entonces es posible aplicarle los parámetros de reverberación de cualquier sala. También podría mejorar enormemente la inteligibilidad de una señal (i.e. un discurso) que fue grabada en un sitio con mucha reverberación.

## ***1.4 OBJETIVOS DE LA INVESTIGACIÓN***

### **1.4.1 OBJETIVO GENERAL**

- Desarrollar un algoritmo de programación capaz de deconvolucionar una señal procesada por un sistema acústico, para encontrar la señal original de entrada al sistema.

### **1.4.2 OBJETIVOS ESPECÍFICOS**

- Optimizar la precisión del algoritmo y el tiempo de procesamiento.
- Evaluar la precisión del algoritmo con respuestas al impulso artificiales basadas en los tiempos de reverberación del sistema.
- Realizar pruebas de deconvolución con sistemas electrónicos.

## ***1.5 ALCANCES Y LIMITACIONES DEL PROYECTO***

### **1.5.1 ALCANCES**

Al final de esta investigación, se pretende desarrollar un algoritmo estable, preciso y confiable para deconvolucionar la señal de salida de un sistema mediante la respuesta al impulso del mismo para obtener la señal de entrada original, con tiempos de procesamiento óptimos y una precisión aceptable en la mayoría de los casos.



### **1.5.2 LIMITACIONES**

Uno de los grandes limitantes del proyecto es que toda la teoría matemática de deconvolución están basados en sistemas LTI, lo cual no es cierto tratándose de sistemas acústicos reales, por lo que el resultado entregado por el algoritmo, siempre será una aproximación a la señal de entrada original del sistema.

## 2 MARCO DE REFERENCIA

### 2.1 MARCO CONCEPTUAL

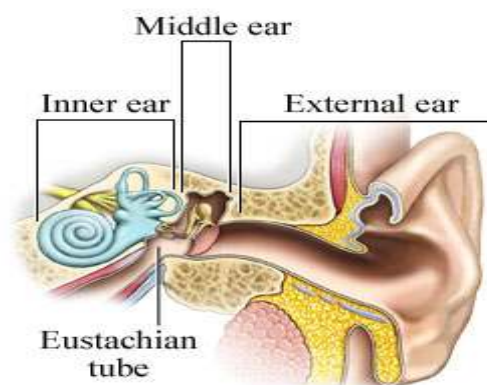
#### 2.1.1 PSICOACÚSTICA

##### 2.1.1.1 Introducción

En este capítulo se hará una breve introducción a la psicoacústica y la anatomía del oído, con el fin de comprender cómo funciona nuestro sistema auditivo y como percibe los estímulos externos. Psicoacústica es un término relativamente nuevo que se refiere a las relaciones funcionales de la anatomía y fisiología del oído con la percepción del sonido. Esto es de gran importancia para entender el por qué de la necesidad de espacios anecoicos y en este caso específico, la necesidad de un algoritmo capaz de deconvolucionar una señal para llevarla a su estado anecoico.

##### 2.1.1.2 Anatomía Del Oído

El oído está compuesto por tres secciones, el oído externo, el oído medio y el oído interno. En la Figura 1 se muestran dichas secciones de forma global.



**Figura 1** Ilustración del oído y los tres segmentos que lo componen; “oído externo:external ear”, “oído medio:middle ear”, “oído interno:inner ear”. Nucleus Medical Art.

#### **2.1.1.2.1 Oído Externo**

El oído externo está conformado por el pabellón u oreja, el conducto externo y el tímpano o membrana timpánica.

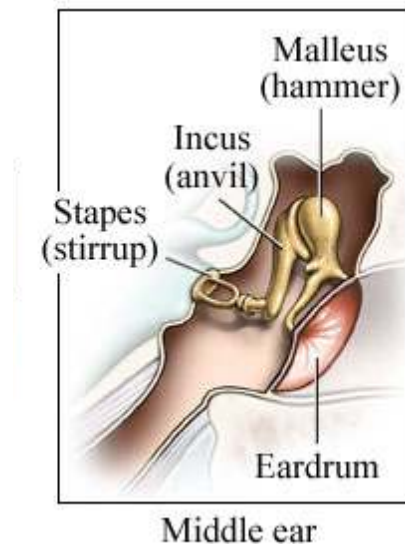
El pabellón dirige las ondas sonoras que inciden sobre él, al conducto externo. Los músculos que controlan el movimiento del pabellón están atrofiados en los seres humanos, a diferencia de otros mamíferos, lo cual limita su función que es ayudar a localizar la proveniencia de un sonido. El conducto externo, como su nombre lo sugiere, conduce las ondas sonoras a través de él y hasta el tímpano. Debido a su forma, constituye físicamente un tubo cerrado en el cual circula aire, por lo que tiene una frecuencia de resonancia propia alrededor de los  $3\text{kHz}$  debido a sus dimensiones ( $2.7\text{cm}$  de largo y un diámetro de  $0.7\text{cm}$ ). Esta resonancia aumenta de  $10\text{dB}$  a  $15\text{dB}$  el nivel de presión sonora, lo cual se manifiesta como una mejora auditiva en un rango de frecuencias que afecta de manera importante la voz (la mayoría de las consonantes no se pueden entender si se elimina el contenido de la señal por arriba de los  $2\text{kHz}$ ). Una vez las ondas llegan al tímpano, este vibra de forma proporcional a como es estimulado.

Debido a que el tímpano tiene forma de cono asimétrico, tiene frecuencias de resonancias propias de estos sistemas, y debido a su forma específica y dimensiones, estas frecuencias de resonancia para el tímpano están entre los  $2\text{kHz}$  y los  $5\text{kHz}$ , contribuyendo con la ganancia del sistema, al igual que el conducto auditivo, en un rango de frecuencias de mucha importancia para la inteligibilidad de la palabra.

#### **2.1.1.2.2 Oído Medio**

El oído medio está conformado por tres estructuras: caja timpánica, trompa de Eustaquio y mastoides. La caja timpánica, que es la única de las tres estructuras involucrada directamente en el proceso de audición (i.e. recepción, transmisión y

percepción), consta de una cadena de huesecillos u osículos y los músculos del oído medio (i.e. MOM) que son: tensor timpánico y músculo del estribo.



**Figura 2** Constitución de la cadena de huesecillos del oído medio; “martillo:hammer”, “yunque:anvil”, “estribo:stirrup”, “eardrum:tímpano”. Nucleus Medical Art.

La cadena de huesecillos se compone de tres pequeños huesos, el martillo que está unido al tímpano, el yunque que conecta al martillo con el estribo, y por último el estribo que está unido a la ventana oval. Esta cadena se encarga de transducir las vibraciones acústicas en vibraciones mecánicas y transmitírselas a la ventana oval, aunque vale la pena aclarar que no toda la energía es transmitida, una parte se pierde en el proceso de transducción debido a la masa, elasticidad y a la fricción ejercida por las uniones de la cadena de huesecillos. Esto se conoce como la impedancia acústica del sistema auditivo. Sin embargo, el sistema auditivo contrarresta este efecto de dos formas: el sistema de palanca conformado por el martillo y el yunque tiene una ganancia mecánica de aproximadamente 1.3 veces la fuerza ejercida inicialmente, y la relación de áreas entre la membrana del tímpano (aprox.  $69\text{mm}^2$ , aunque su área efectiva de vibración es de aprox.  $43\text{mm}^2$ ) y la de la ventana oval (aprox.  $3\text{mm}^2$ ) es de 14:1, lo cual implica un acople de impedancia debido a que la fuerza es amplificada entre el martillo y el yunque y el área disminuye con respecto al área inicial, resultando en un aumento de la presión (fuerza por unidad de área) final.

Los MOM, por otro lado, se contraen para limitar el movimiento de los huesecillos (i.e. aumentan la rigidez del sistema) y así proteger el oído de posibles daños debido a estímulos acústicos muy severos. Aún así, estos músculos tienen una latencia de respuesta con respecto al estímulo, usualmente entre  $10ms$  y  $15ms$ , por lo que no ayudan mucho a proteger el oído de ruidos de tipo impulsivo<sup>1</sup>. A pesar de esta latencia, estos músculos pueden contraerse de forma anticipada a la generación del ruido impulsivo si el escucha está predispuesto o conoce de antemano que el ruido será generado. Starr (1969), demostró que la contracción de los MOM aumentan la transmisión de energía a la cóclea para frecuencias entre  $1kHz$  y  $2kHz$ , atenuando todas las frecuencias fuera de este rango.

### ***2.1.1.2.3 Oído Interno***

El oído interno está conformado por la ventana oval, la cóclea o laberinto anterior y la ventana redonda<sup>2</sup>. Su función principal es transducir la energía mecánica transmitida por el oído medio en señales eléctricas que el cerebro pueda interpretar. La ventana oval, conectada a uno de los huesos de la cadena de huesecillos (i.e. el estribo), transmite la vibración mecánica que le llega a esta a la cóclea.

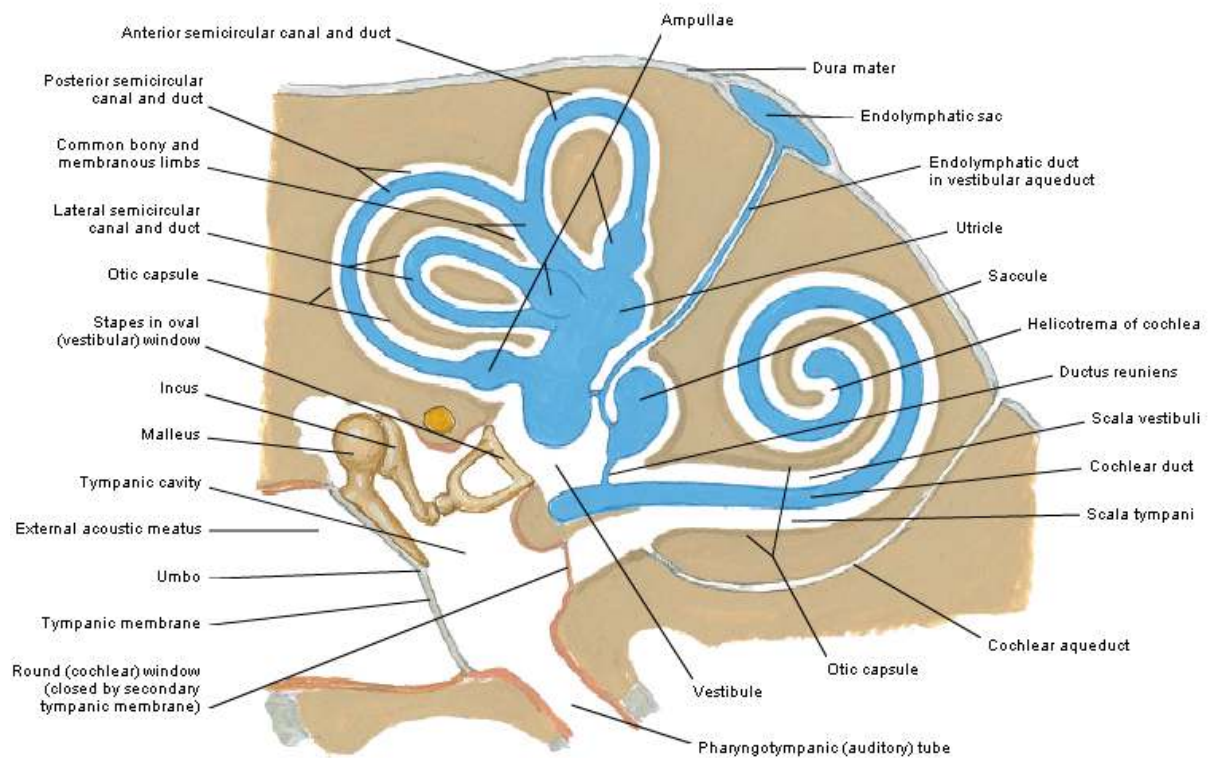
La cóclea es un conducto de aproximadamente  $33mm$  de largo en el hueso temporal en forma de espiral o caracol, cubierto de un laberinto membranoso que se extiende a través de toda la cóclea y que a su vez divide la cóclea en tres rampas o conductos: rampa vestibular, rampa timpánica y rampa coclear. La rampa vestibular y la rampa timpánica están llenas de un líquido llamado perilinfa el cual vibra estimulado por la ventana oval. Ambas rampas se encuentran en el vértice de la cóclea o helicotrema. La perilinfa, es un líquido incompresible<sup>3</sup>, por lo que la ventana oval y la ventana redonda siempre se tendrán un desfase de  $180^\circ$  la una con respecto a la otra.

---

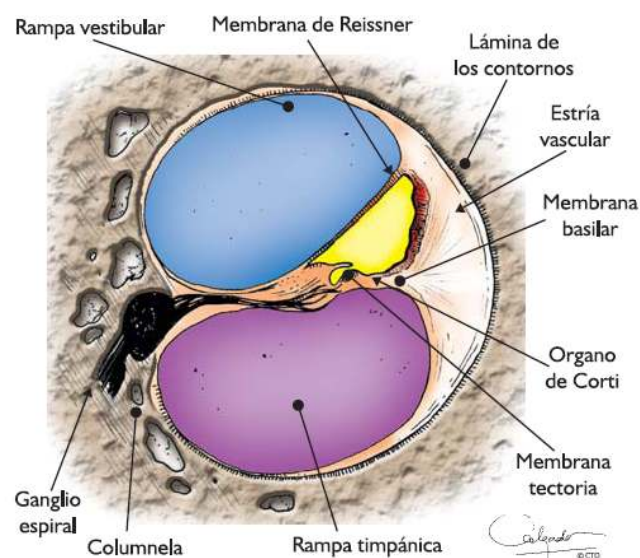
<sup>1</sup> Se entiende por ruido impulsivo, aquel cuya nivel aumenta en más de 35dB en menos de 1 segundo (i.e. una explosión).

<sup>2</sup> El oído interno también está compuesto por el sistema de semicírculos o laberinto posterior, que son relevantes para la audición.

<sup>3</sup> Propiedad de algunos fluidos. El fluido no deja comprimir o reducir su volumen.



**Figura 3** Conexión del oído medio con el oído interno y detalles de los laberintos membranosos de este último. Frank Netter.



**Figura 4** Laberinto membranoso y su estructura dentro de la cóclea. Mini-manual del otorrino, <http://www.redsaludchepen.gob.pe/publicaciones/>.

La rampa coclear, ubicada entre la rampa vestibular y la rampa timpánica y separada de estas por la membrana de Reissner o vestibular y la membrana basilar respectivamente, se encuentra llena de un líquido llamado endolinfa que debido a la diferencia en su composición química con respecto a la composición de la perilinfa genera una diferencia de potencial en la membrana basilar. Esto no sucede con la membrana de Reissner ya que esta es muy delgada, por lo que los líquidos podrían considerarse en la misma rampa, pero separados por su diferencia de densidades. El movimiento de la perilinfa en la rampa vestibular, genera un movimiento en la endolinfa, que a su vez hace vibrar la membrana basilar. El punto de deformación máxima o punto de resonancia de la membrana basilar, dependerá de la frecuencia del estímulo; mientras menor sea la frecuencia del estímulo, la membrana tendrá un punto de resonancia más alejado de la ventana oval y más cerca de la helicotrema y ocurre lo contrario para las frecuencias altas, mientras más alta la frecuencia, más cerca de la ventana oval se producirá la excitación de la membrana basilar. Esto es debido a que la membrana basilar no tiene la misma rigidez en toda su longitud, siendo más rígida en la base de la cóclea y perdiendo rigidez a medida que se acerca al helicotrema. Es por esto que percibimos la altura o nivel de un sonido; el cerebro procesa por separado las frecuencias del estímulo, actuando como un analizador espectral por filtros.

Esta vibración en la membrana basilar, excita las células ciliadas en el órgano de Corti (ubicado sobre la membrana basilar), generando impulsos eléctricos que dependerán del tiempo del estímulo y de la deformación de estas células. Estos impulsos eléctricos serán procesados por el cerebro, para luego ser interpretados como sonido.

### **2.1.1.3 Niveles De Presión Y El Decibel**

El oído humano es capaz de percibir cambios en la presión atmosférica desde los  $20\mu Pa$  (i.e. umbral de audición) hasta los  $20Pa$  (i.e. umbral del dolor); la diferencia de presión entre los dos umbrales es de 1000000:1, lo cual nos da una idea del fantástico

rango dinámico que posee nuestro oído. Sin embargo, esto representa un problema cuando se manipulan datos matemáticamente, ya que resultaría muy incómodo manejar valores tan diferentes dentro de una misma escala. Por esta razón, se utiliza como medida de presión sonora el decibel. El decibel es una relación logarítmica entre dos cantidades<sup>4</sup>, en el caso específico del sonido, es la relación entre dos perturbaciones a la presión atmosférica. Un decibel es la décima parte de un Bel<sup>5</sup>. La siguiente expresión matemática (0.1) describe la relación en decibeles entre dos potencias:

$$dB = 10 \log_{10} \left( \frac{P_1}{P_2} \right) \quad (0.1)$$

Donde  $P_1$  y  $P_2$  son los valores de potencia. Debido a que el decibel es una relación entre dos cantidades, se estandarizó referenciar los  $L_p$  (i.e. niveles de presión sonora) al umbral mínimo de audición, es decir  $20 \mu Pa$ , con el propósito de tener una referencia y una medida de niveles de presión sonora universal. La ecuación (0.2) muestra la definición matemática del  $L_p$ :

$$L_p = 10 \log_{10} \left( \frac{p^2}{p_{ref}^2} \right) \quad (0.2)$$

Donde  $p^2$  es la presión *RMS* al cuadrado o potencia acústica a referenciar y  $p_{ref}^2$  el la potencia de referencia.

La ecuación (0.2) se puede simplificar como se muestra en la ecuación (0.3).

$$L_p = 20 \log_{10} \left( \frac{p}{p_{ref}} \right) \quad (0.3)$$

---

<sup>4</sup> El decibel también se usa para describir relaciones entre voltajes, potencias y otras cantidades físicas.

<sup>5</sup> Medida inventada por los ingenieros del Bell Telephone Laboratory para calcular la pérdida por transmisión de una señal de audio en una milla de cable estándar para líneas telefónicas.



En la ecuación (0.3) se puede ver que la relación de las dos presiones solo se hará igual a uno cuando  $p = p_{ref}$ , es decir, cuando  $p = 20\mu Pa$ . Cuando esto sucede  $L_p = 0dB$ , que es el umbral de audición en decibeles.

Manejar los niveles de presión sonora por medio del decibel presenta otras ventajas interpretativas. Por ejemplo, es aproximadamente el mínimo cambio en los niveles de presión detectable por el oído humano, es decir, que un sonido ha de aumentar o disminuir sus niveles en al menos  $1dB$  para que el cambio sea perceptible por nuestro oído.

La siguiente tabla comparativa, nos ilustra la diferencia entre los valores de presión sonora y los valores SPL.

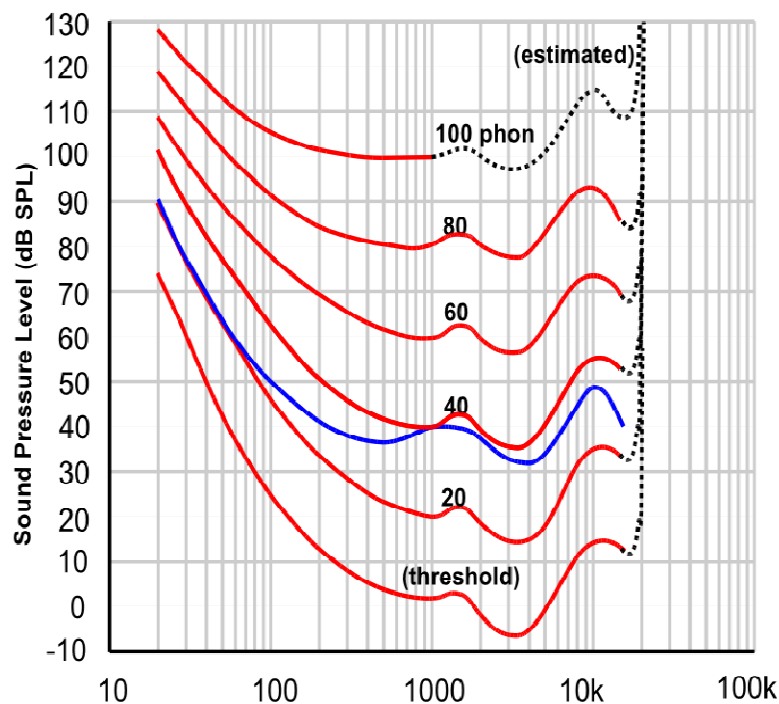
Límite teórico de presión sonora	$101325Pa$ o $1atm$	$194dB$
Erupción volcán Krakatoa @ 160Km	$20000Pa$	$180dB$
Motor de un jet @ 30m	$630Pa$	$150dB$
Disparo de un rifle @ 1m	$200Pa$	$140dB$
Umbral del dolor	$100Pa$	$134dB$
Umbral de molestia	$20Pa$	$120dB$
Paso de un jet @ 100m	$6 - 200Pa$	$110 - 140dB$
Martillo neumático @ 1m	$2Pa$	$\approx 100dB$
Avenida principal @ 10m	$2 \times 10^{-1} - 6 \times 10^{-1} Pa$	$80 - 90dB$
Paso de un auto @ 10m	$2 \times 10^{-2} - 2 \times 10^{-1} Pa$	$60 - 80dB$
Conversación normal @ 1m	$2 \times 10^{-3} - 2 \times 10^{-2} Pa$	$40 - 60dB$
Respiración humana @ 1m	$6 \times 10^{-5} Pa$	$10dB$
Umbral de audición @ 2kHz	$2 \times 10^{-5} Pa$	$0dB$

**Tabla 1** Relación de presión sonora en pascals y niveles de presión sonora en decibeles.  
[http://en.wikipedia.org/wiki/Sound\\_pressure](http://en.wikipedia.org/wiki/Sound_pressure).

#### 2.1.1.4 Percepción Subjetiva De Niveles Sonoros

El oído humano no percibe los estímulos acústicos por igual. La respuesta del sistema auditivo ante un estímulo cualquiera dependerá de muchos factores, como la duración, frecuencia, nivel, etc.. La percepción subjetiva de niveles sonoros o sonoridad de ahora en adelante, trata específicamente los factores que afectan el nivel aparente de un sonido respecto a un oyente.

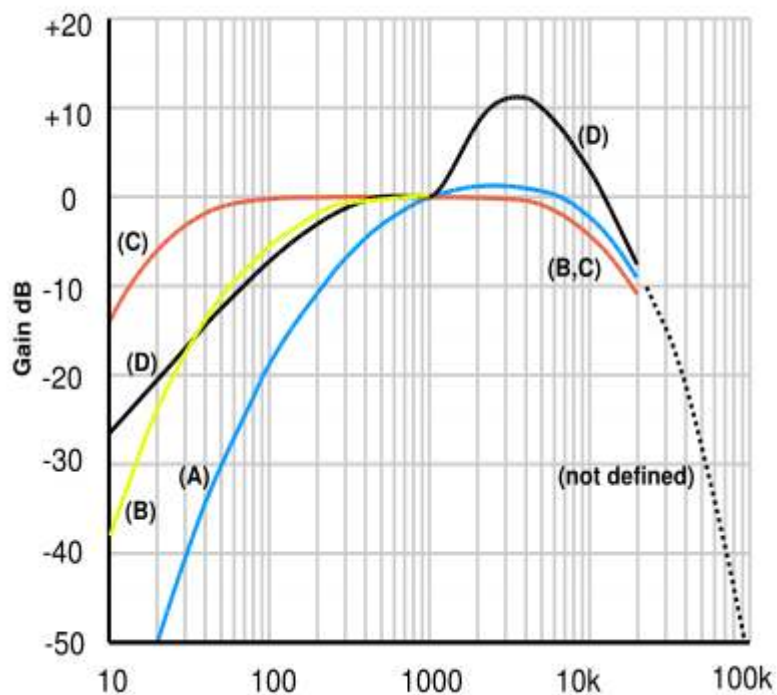
Debido a que la sonoridad percibida por un receptor varía de individuo a individuo, solo se puede plantear una unidad de medida estadística, haciendo énfasis en las características más comunes del oído. En la figura 5 podemos ver las curvas de igual sonoridad donde se ilustra la relación entre los niveles de presión sonora reales y los niveles percibidos por el oído. Como se puede observar en la figura, el oído presenta una mayor sensibilidad alrededor de los 2kHz, lo que resulta conveniente ya que es un rango de frecuencias de gran importancia para el habla y la música.



**Figura 5** Curvas de igual sonoridad ISO revisión 2003 (rojo) y curva de igual sonoridad ISO original para 40 fones (azul). <http://en.wikipedia.org/wiki/Loudness>

### 2.1.1.5 Curvas De Ponderación

Debido a la respuesta del oído ante diferentes estímulos de diferentes frecuencias, los SPL no son por si mismos información útil para cuantificar la sonoridad de un estímulo. Es por esto que se usan diferentes curvas de ponderación que ayudan a evaluar la percepción subjetiva de un sonido. Por ejemplo, ubiquemos la curva de 20 fones en la Figura 5 y se puede observa que para una frecuencia de 20Hz se necesitaría de un SPL 70dB por encima (i.e. 10'000,000 de veces más potencia sonora) del SPL de una frecuencia de 1kHz para generar la misma sonoridad. Por esto, cuando se quiere evaluar la percepción subjetiva, es necesario un mecanismo de corrección.



**Figura 6** Curvas de ponderación de igual sonoridad. Ponderación A (azul), ponderación B (amarillo), ponderación C (rojo) y ponderación D ((negro). <http://en.wikipedia.org/wiki/Weighting>.

En la Figura 6 se muestran diferentes curvas de ponderación para diferentes usos: La curva A, está basada en la curva de los 40 fones, por lo que es útil para medir bajos niveles de presión sonora, como ruido de fondo; Las curvas B y la C están diseñadas para acomodarse a la respuesta de oído cuando es sometido a SPL más altos (alrededor

de los 90 fones); La curva D fue especialmente diseñada para medir la percepción subjetiva del ruido causado por aeronaves. Si miramos la curva A, se hace notorio que para 20Hz serían necesarios 50dB por encima del SPL de una frecuencia de 1kHz para que la curva de el mismo valor.

Aunque estas curvas son de gran utilidad, podemos decir que no se ajustan exactamente a la respuesta del oído, por lo que la precisión de la corrección dependerá de las condiciones de la medición, por ejemplo, si una fuente acústica fluctúa entre 40dB y 100dB en cortos períodos de tiempo, ninguna de las curvas anteriores se acomodaría, por lo que sería necesaria una curva de ponderación adaptativa o simplemente escoger la curva de mayor predominancia. Para diferenciar cuando un nivel sonoro ha sido medido o está planteado bajo una curva de ponderación se agrega la letra de la curva después de la abreviación dB (i.e. 80dB-A, 80dB(C), 80dB(D)).

## **2.1.2 ACÚSTICA EN RECINTOS CERRADOS**

### **2.1.2.1 Introducción**

Los fenómenos acústicos en un recinto cerrado tienen repercusiones importantes sobre la forma en que percibimos el habla, la música y el sonido en general. Por esto, las condiciones acústicas de un recinto deberán cambiar, dependiendo del propósito del recinto y del receptor. Para este fin, se acondiciona acústicamente el recinto para moldear sus características. La deconvolución, a pesar de no ser un acondicionamiento acústico<sup>1</sup>, regresa la señal captada en un recinto a su estado original o anecoico, lo cual nos permite mediante procesos digitales, procesar esta nueva señal y acomodarla a nuestras necesidades.

### **2.1.2.2 Reflexiones Acústicas**

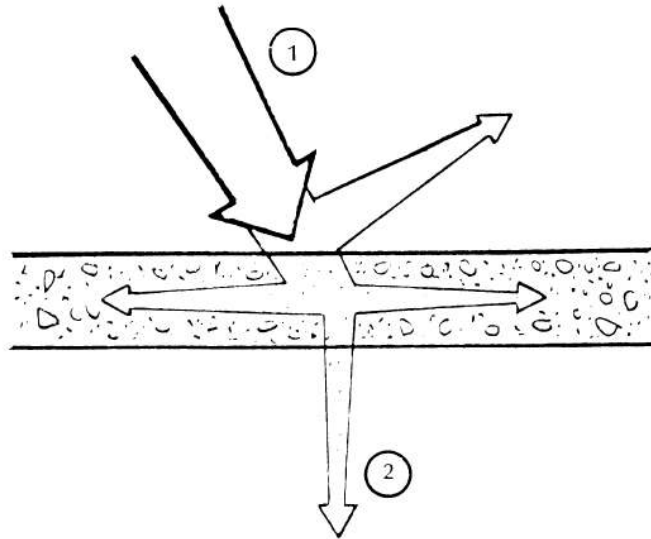
Cuando una onda sonora encuentra un obstáculo en su camino ocurren varios fenómenos. Parte de energía de la onda que choca contra el obstáculo se refleja, parte es absorbida y otra parte se transmite a través de este. La Figura 7 nos muestra estos tres fenómenos de los cuales nos centraremos en dos: la absorción y la reflexión.

Las características del material (i.e. porosidad, rugosidad, forma de la superficie, etc.) del obstáculo definirán cuanta energía de la onda incidente será absorbida por este y disipada en forma de calor. La energía que no es absorbida (o transmitida) será reflejada. Para relacionar ambas propiedades del material, su capacidad de absorción y su capacidad reflectiva, se utiliza un solo parámetro: el coeficiente de absorción, que es

---

<sup>1</sup> Mediante la acústica asistida, la deconvolución podría considerarse como un acondicionamiento acústico activo.

la relación de la energía reflejada sobre la energía incidente y está dada por la ecuación (0.4).



**Figura 7** Reflexión, absorción y transmisión de una onda sonora que incide sobre un obstáculo. Manual del aislamiento en edificaciones, ISOVER.

$$\alpha = 1 - \frac{P_r}{P_i} \quad (0.4)$$

Donde  $P_r$  es la energía reflejada y  $P_i$  es la energía que incide sobre el material y  $P_r/P_i$  es el coeficiente de reflexión. Ya que la energía reflejada nunca será mayor que la energía incidente<sup>2</sup>, el coeficiente de absorción siempre estará entre valores de 0 a 1, donde cero significa absorción nula (i.e. reflectividad total) y 1 sería absorción total (i.e. reflectividad nula). El coeficiente de absorción varía en función de la frecuencia de la onda incidente (ver Tabla 2).

---

<sup>2</sup>

	Coeficientes de absorción					
	125Hz	250Hz	500Hz	1kHz	2kHz	4kHz
Bloque de hormigón tosco	0.36	0.44	0.31	0.29	0.39	0.25
Bloque de hormigón pintado	0.10	0.05	0.06	0.07	0.09	0.08
Cristal ordinario de ventana	0.29	0.10	0.05	0.04	0.07	0.09
La superficie del agua, como en una piscina	0.01	0.01	0.01	0.02	0.02	0.03
Lana de vidrio de 2.54cm, 24kg a 48kg/m <sup>3</sup>	0.08	0.25	0.65	0.85	0.80	0.75
Lana de vidrio de 5,1cm, 24kg a 48kg/m <sup>3</sup>	0.17	0.55	0.80	0.90	0.85	0.80

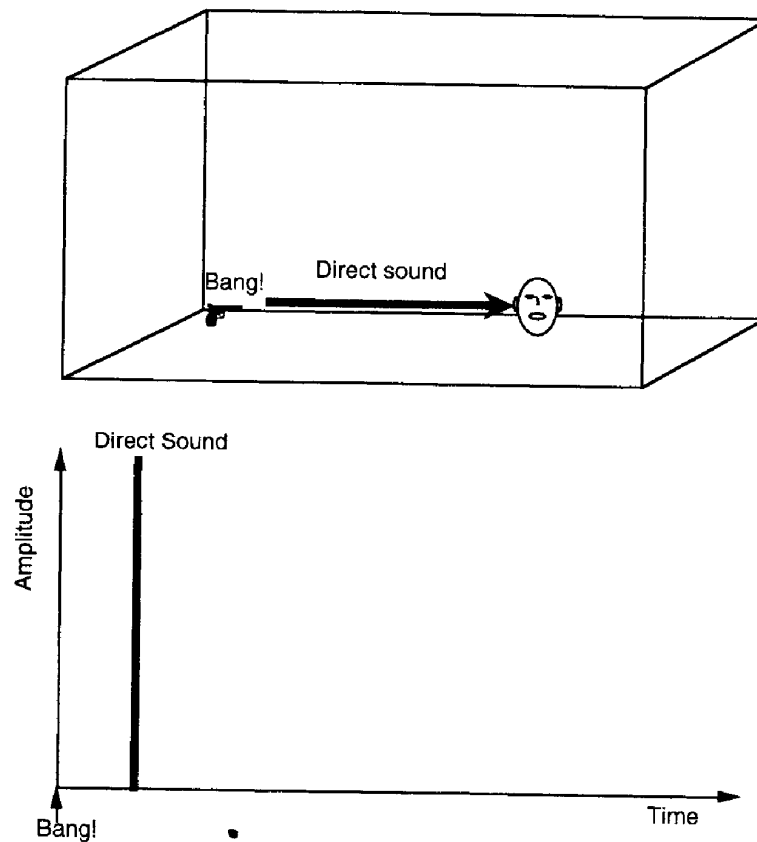
**Tabla 2** Coeficientes de absorción para algunos materiales comunes. Cyril M. Harris, Manual de medidas acústicas y control del ruido, McGraw Hill.

En la Tabla 2 podemos observar que materiales bastante rígidos, con densidades volumétricas altas como el cristal ordinario o el agua, son altamente reflectivos, mientras materiales porosos y de superficies rugosas como la lana de vidrio son altamente absorbentes. Un caso especial es el del bloque de hormigón tosco y el mismo bloque pintado, donde nos damos cuenta que el bloque pintado, al ser revestido con la pintura, pierde porosidad desde el punto de vista de la onda incidente y pasa a ser un material más reflectivo que el hormigón tosco.

Una observación general es que las frecuencias altas tienden a ser más fáciles de atenuar o absorber, dado que sus longitudes de onda son comparables con las dimensiones del obstáculo, en cambio, para absorber una frecuencia baja (i.e. la longitud de onda de una onda con 100Hz de frecuencia es de 3.4m) se utilizan otras técnicas, generalmente mecánicas, para atenuar la energía incidente sobre el obstáculo.

### 2.1.2.3 Reflexiones Acústicas en Recintos Cerrados

Ubiquémonos en la siguiente situación. Se genera un sonido impulsivo (i.e. se dispara un arma de fuego) dentro un recinto y hay un receptor ubicado como aparece en la Figura 8. Podemos describir lo que sucede en tres etapas: Sonido directo, reflexiones tempranas y campo reverberante.



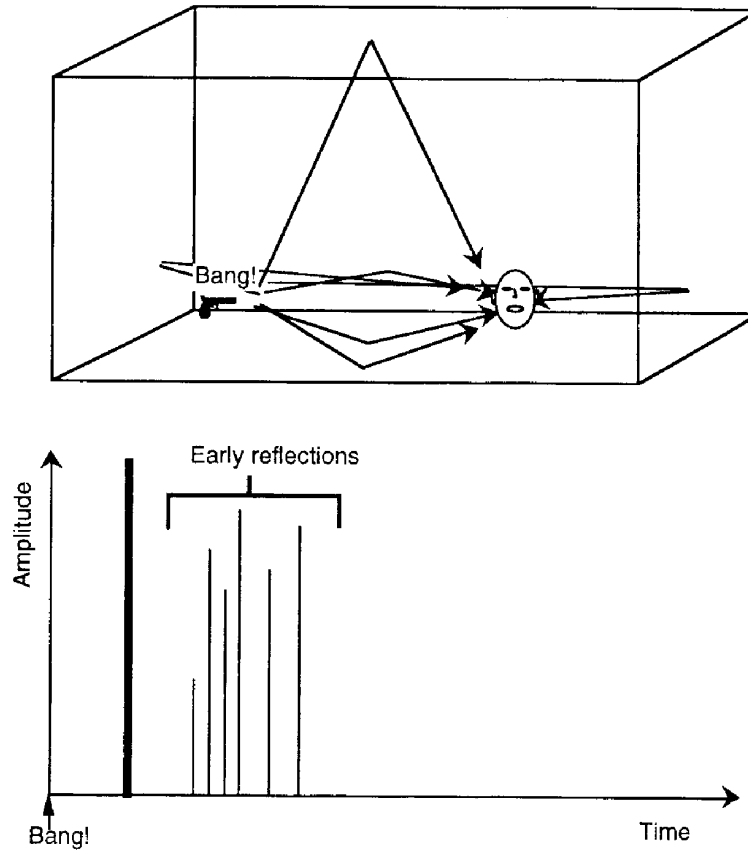
**Figura 8** Sonido directo.  
[http://www.postaudio.co.uk/education/acoustics/room\\_acoustics.html](http://www.postaudio.co.uk/education/acoustics/room_acoustics.html).

El sonido directo es un frente de onda que viaja directamente de la fuente al receptor y por esto no está alterado de ninguna forma por el recinto, por lo que se propaga de igual manera que una onda campo libre como se muestra en la ecuación (0.5), donde  $L_1$  y  $d_1$  son los niveles y distancia de referencia respectivamente y  $L_2$  es el nivel a una distancia  $d_2$  de la fuente. Si la distancia fuente-receptor fuera de 10m, la demora de arribo de este primer frente de onda, sería de unos 3ms (asumiendo una velocidad de propagación del sonido de 344m/s)<sup>3</sup> en llegar al receptor. Esta este primer frente de onda es el más importante para el oído en cuanto a la localización de la fuente acústica.

<sup>3</sup> Velocidad aproximada de propagación del sonido a 20°C temperatura ambiente.

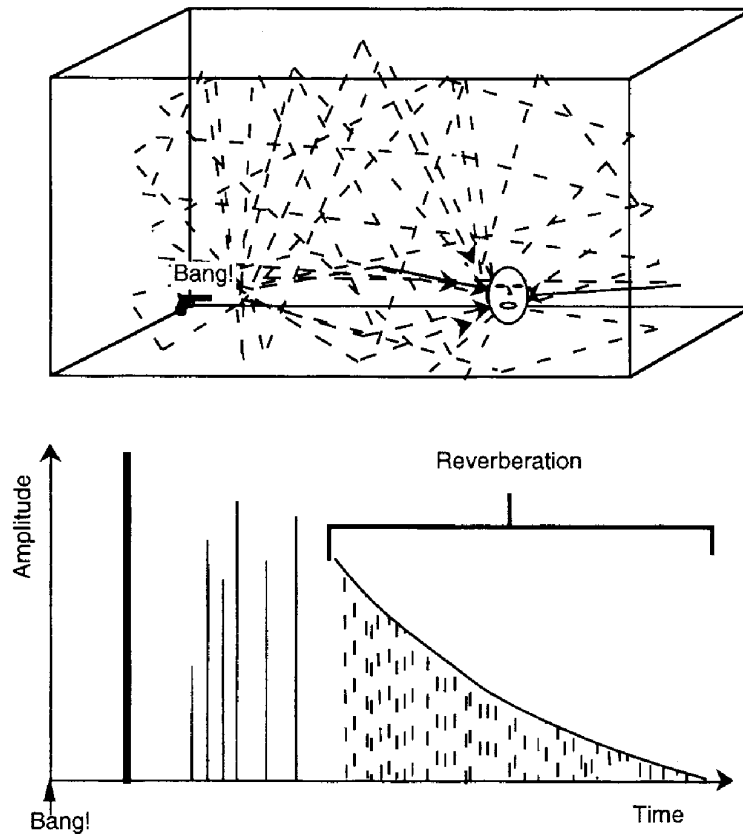


$$L_2 = L_1 - 20 \log \frac{d_2}{d_1} \quad (0.5)$$



**Figura 9** Reflexiones tempranas.  
[http://www.postaudio.co.uk/education/acoustics/room\\_acoustics.html](http://www.postaudio.co.uk/education/acoustics/room_acoustics.html).

Luego del sonido directo, llegan las primeras reflexiones al receptor como se ve en la Figura 9. Estas corresponden a los caminos más cortos, después del recorrido del sonido directo, hasta el receptor y llegan en tiempos discretos, una a la vez, por lo que pueden analizarse por separado y se puede estimar la proveniencia de cada una. Las reflexiones tempranas varían dependiendo de la posición del receptor y de la fuente en el recinto. Dichas reflexiones tiene gran influencia sobre la percepción tímbrica del sonido y pueden alterar el sentido de localización de la fuente acústica si llegan dentro de los primeros 25-30ms después de percibido el sonido directo.

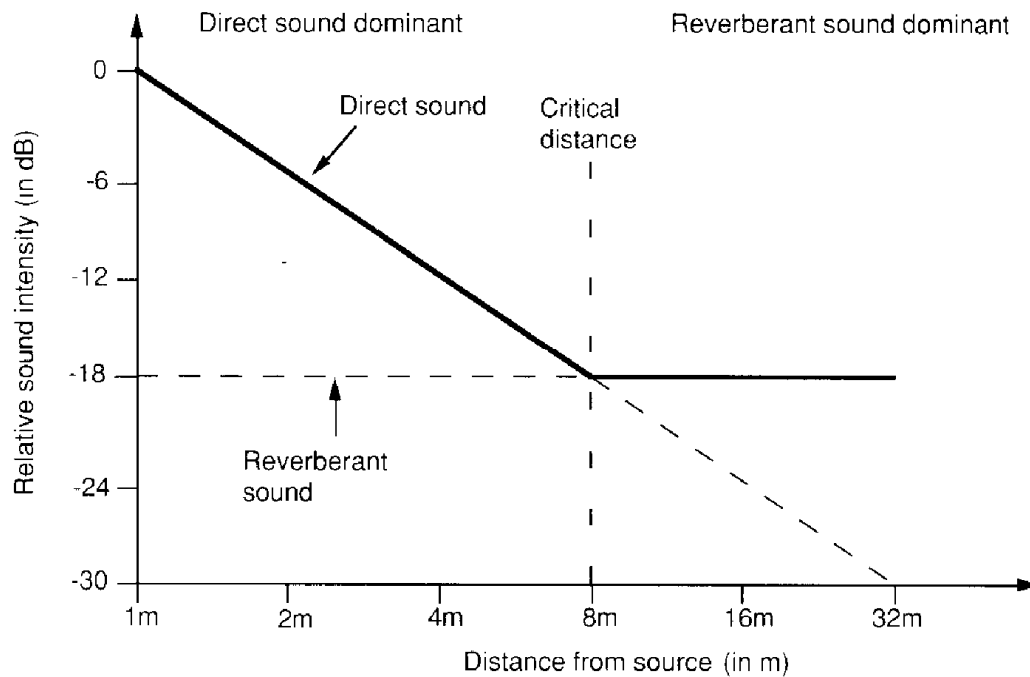


**Figura 10** Reverberación.  
[http://www.postaudio.co.uk/education/acoustics/room\\_acoustics.html](http://www.postaudio.co.uk/education/acoustics/room_acoustics.html).

Finalmente, comienzan a llegar reflexiones tardías, provenientes de todas las superficies al mismo tiempo, las cuales constituyen el campo reverberante. El tiempo de reverberación es el tiempo que estas reflexiones demoran ser absorbidas totalmente por el recinto. A diferencia de las reflexiones tempranas, las reflexiones que constituyen el campo reverberante no pueden ser analizadas por separado y por esto no es posible saber su procedencia exacta, por lo que se establece que existe la misma probabilidad de que arriben en cualquier dirección y de que provengan de cualquier superficie.

La Figura 11 nos ilustra la relación entre el campo directo y el campo reverberante. La distancia crítica, es la distancia a partir de la cual comienza a influir el campo reverberante sobre el campo directo. A medida que nos alejamos de la distancia crítica,

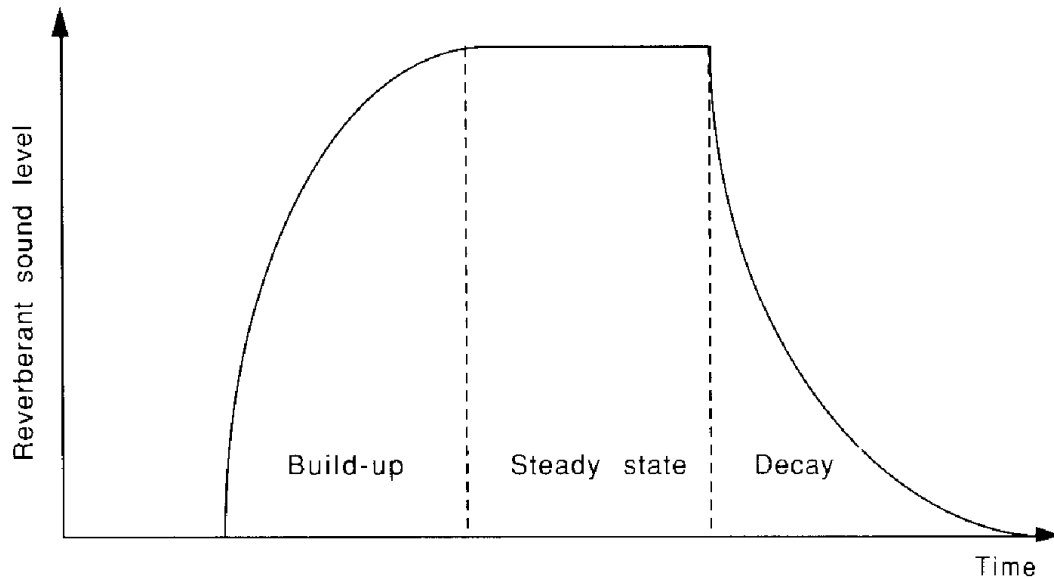
tendrá más influencia el campo reverberante sobre los niveles percibidos que el campo directo.



**Figura 11** Relación de la distancia crítica con el campo directo y el campo reverberante.  
[http://www.postaudio.co.uk/education/acoustics/room\\_acoustics.html](http://www.postaudio.co.uk/education/acoustics/room_acoustics.html).

Una de las formas de calcular los tiempos de reverberación de un recinto es por medio de la ecuación de Sabine (ver ecuación (0.6)) que evalúa la cantidad de tiempo necesaria para que el recinto absorba 60dB de la energía inicial del campo reverberante. Cómo podemos ver es esta expresión, el tiempo de reverberación es directamente proporcional al volumen del recinto  $V$  e inversamente proporcional a la absorción promedio  $\alpha$  y la suma de sus superficies  $S$ .

$$RT_{60} = \frac{0.161V}{S\alpha} \quad (0.6)$$



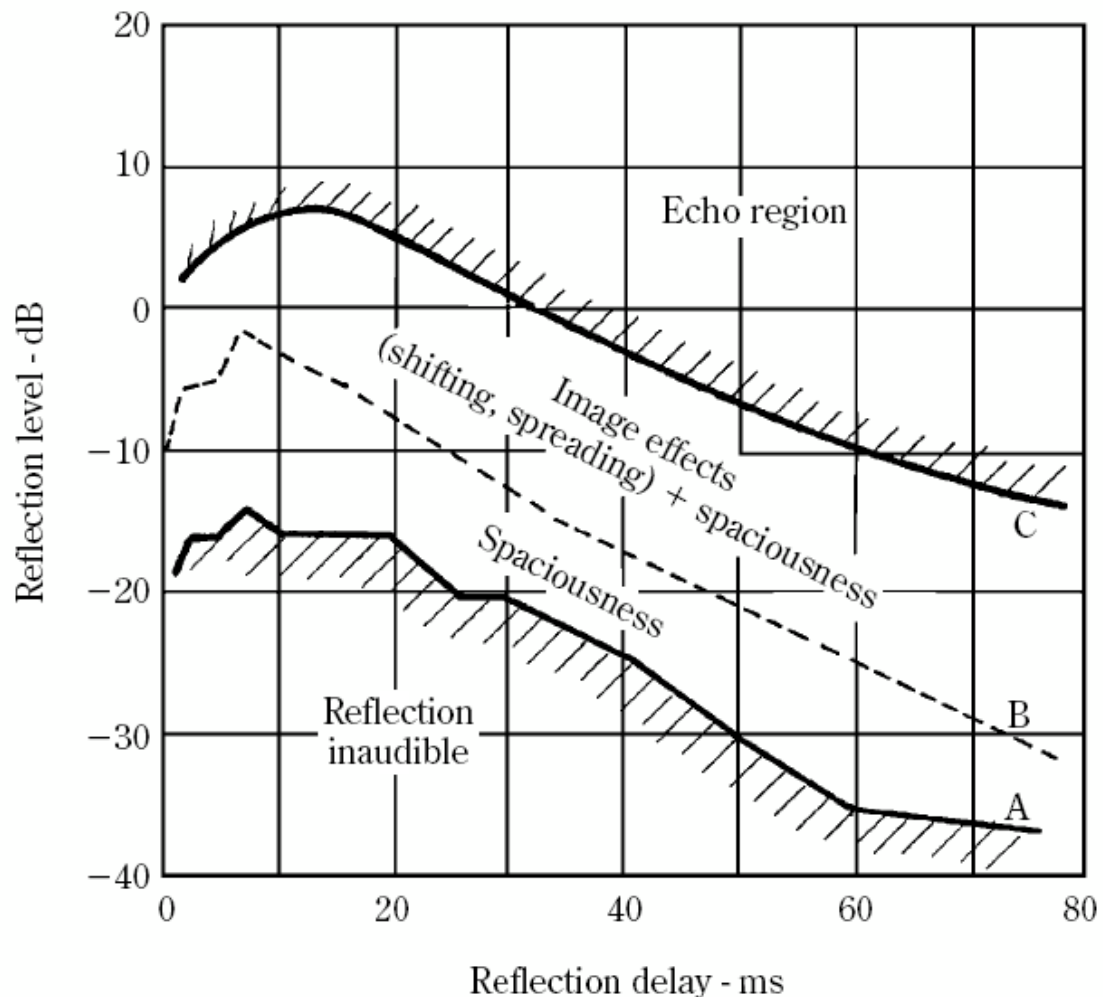
**Figura 12** Reverberación.  
[http://www.postaudio.co.uk/education/acoustics/room\\_acoustics.html](http://www.postaudio.co.uk/education/acoustics/room_acoustics.html).

El nivel de la reverberación en función del tiempo se muestra en la Figura 12, donde podemos ver que este tarda un tiempo en estabilizarse y luego decae exponencialmente una vez la fuente ha cesado. Una vez estabilizado el campo reverberante, se puede decir que la energía está uniformemente distribuida en todo el recinto.

#### **2.1.2.4 Efectos De Las Reflexiones Tempranas**

En la situación de la Figura 9, concentrémonos en el sonido directo y la primera reflexión que llega al receptor. Los efectos de esta primera reflexión dependerán de cómo y cuando llegó esta al escucha en relación con el sonido directo. La Figura 13 muestra los resultados de los estudios realizados por Olive y Toole (ver

BIBLIOGRAFÍA) donde se pueden apreciar estos efectos y los factores que influyen en ellos.



**Figura 13** Efectos de una reflexión contra el tiempo de arribo y su nivel después del sonido directo.  
F. Alton Everest, Master handbook of acoustics, McGraw Hill.

La curva A es el umbral de percepción de la señal retrasada, es decir, por debajo de esta curva la reflexión es imperceptible por el oído. Se puede observar que el oído se vuelve más sensible a las reflexiones a medida que el retraso de arribo de estas aumenta. Cuando la señal retrasada se encuentra entre la curva A y la curva B, el oído interpreta la reflexión como un cambio en la espacialidad del recinto (i.e. cambio en las dimensiones percibidas o aparentes del sitio) y un aumento en el nivel sonoro, sin poder identificar la reflexión como un evento discreto o aparte del sonido directo. Este

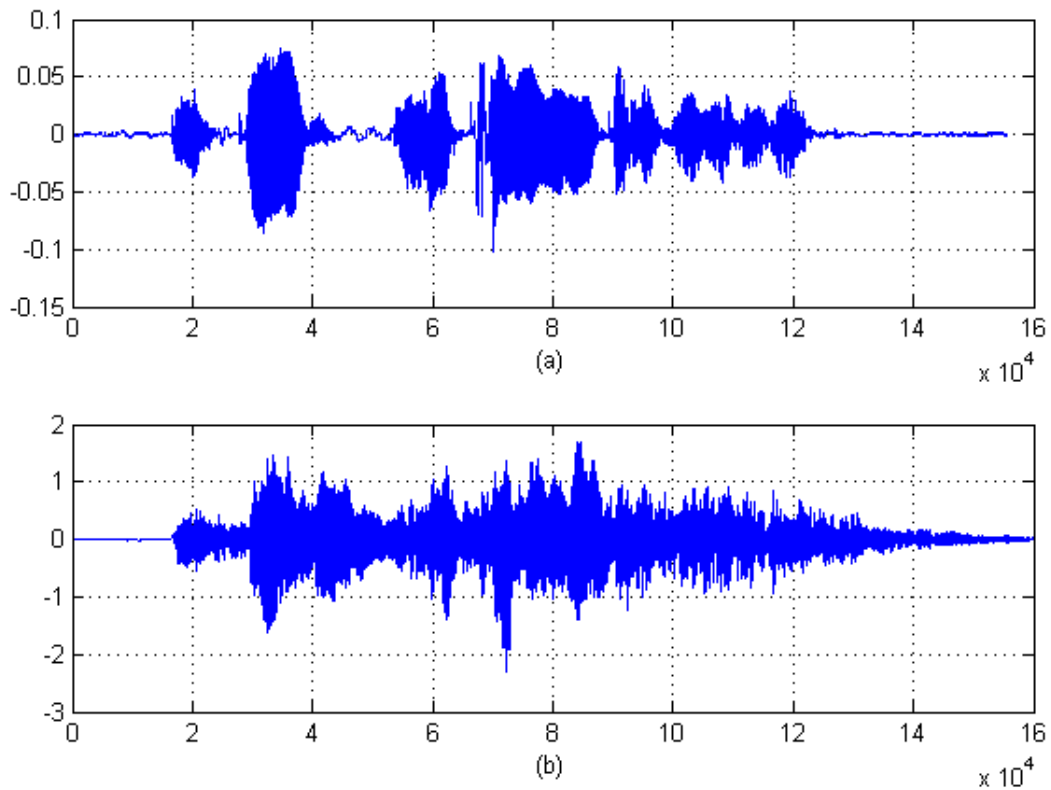
aumento o modificación en la espacialidad del recinto resulta en general favorable, según Haas: “es una placentera modificación de la impresión sonora, haciendo más amplio el sonido directo, mientras que la fuente del eco no es percibida acústicamente”.

Entre la curvas B y C, comienzan a aparecer los problemas o efectos no deseados de las reflexiones tempranas, los cambios de la imagen acústica. La imagen acústica es generada por el oído para recrear un evento acústico en el cerebro, como la posición y tamaño de la fuente generadora. A medida que el retraso de la señal reflejada aumenta, la posición aparente de la fuente acústica se inclina más hacia la proveniencia de la señal reflejada, lo que genera una sensación de confusión para el receptor.

Por encima de la curva C, las reflexiones son percibidas como ecos discretos, y esto afecta gravemente la calidad sonora percibida por el receptor y es por esto que en recintos de escucha crítica dichos ecos deben ser controlados sin excepciones.

#### **2.1.2.5 Efectos Del Campo Reverberante**

El campo reverberante tiene una gran influencia en la percepción del sonido dentro de un recinto. Pero lo que define si el efecto del campo reverberante es positivo o negativo es el uso o propósito del recinto. Por ejemplo, en un auditorio diseñado para discursos hablados, los tiempos de reverberación tienen que ser muy bajos, ya que la inteligibilidad de la palabra es inversamente proporcional al tiempo de reverberación del sitio. Para esclarecer esto, la Figura 14 muestra la frase “Esto es una prueba de inteligibilidad” pronunciada en un recinto “muerto” (i.e. tiempos de reverberación bajos) y en un recinto “vivo” (i.e. tiempos de reverberación altos). Es notable como en el recinto reverberante, se enmascaran las palabras entre si al ser influenciadas por el campo reverberante del recinto, por lo que la inteligibilidad se ve afectada negativamente.



**Figura 14** Arriba (a), varias palabras dichas dentro de un recinto con poca reverberación y abajo (b) las mismas palabras pronunciadas en un recinto con una reverberación considerablemente mayor.

Sin embargo, para el caso de un recinto diseñado para escuchar música, la reverberación puede ayudar a mejorar la experiencia del oyente. Los parámetros de medición de la calidad acústica de un recinto se basan en la experiencia adquirida en salas de gran prestigio por su buena acústica según su uso.

$$RT_{mid} = \frac{RT(500Hz) + RT(1kHz)}{2} \quad (0.7)$$

La ecuación (0.7) nos muestra un parámetro muy utilizado para el diseño de recintos acústicos de escucha crítica. El  $RT_{mid}$  es el promedio de los tiempos de reverberación en 500 y en 1000Hz. Según Beranek, las salas para conciertos sinfónicos deben tener un  $RT_{mid}$  entre 1.8 y 2 segundos para obtener una buena calidad sonora mientras que las

salas para ópera deben tener un  $RT_{mid}$  entre 1.2 y 1.5 segundos. Obviamente, la sala para conciertos sinfónicos, las fuentes acústicas son generalmente instrumentos acústicos que se ven beneficiados por una alta reverberación debido a que el sonido proveniente de todos los instrumentos conforma un campo acústico uniforme con el campo reverberado, mientras que para la ópera, siendo la principal fuente la voz de un intérprete, se necesita un compromiso entre la inteligibilidad y la calidad musical.

Como podemos ver, los efectos de la reverberación en la música son más difíciles de estudiar ya que están basados en la percepción subjetiva, pero su importancia en la experiencia del escucha es decisiva.

## **2.1.3 SEÑALES Y SISTEMAS DISCRETOS**

### **2.1.3.1 Introducción**

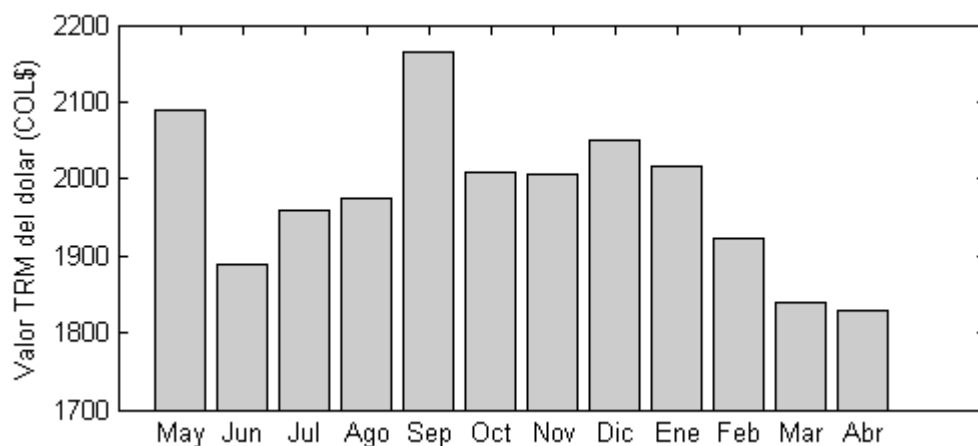
La deconvolución es un proceso matemático, basado en la teoría de sistemas invertibles LTI, que nos permite devolver una señal captada como salida de un sistema (i.e. señal de un discurso grabada en un auditorio) a su estado original sin reflexiones o anecoico. En este capítulo se tratarán todas las bases de procesamiento de señales y sistemas necesarios para finalmente entender el concepto de deconvolución. Solo trataremos las señales y sistemas discretos, ya que en esta investigación, la deconvolución se logrará por medio de un algoritmo de programación digital que constituye un sistema discreto en si. También se tratarán en este capítulo los conceptos matemáticos de las diferentes aproximaciones al algoritmo de deconvolución, lo cual luego, nos permitirá analizar las ventajas y desventajas de cada uno de los métodos.



Los sistemas reales acústicos nunca son perfectamente lineales o invariantes en el tiempo, por esto, dedicaremos una sección del capítulo a hablar del comportamiento real de los sistemas acústicos.

### 2.1.3.2 Señales Discretas

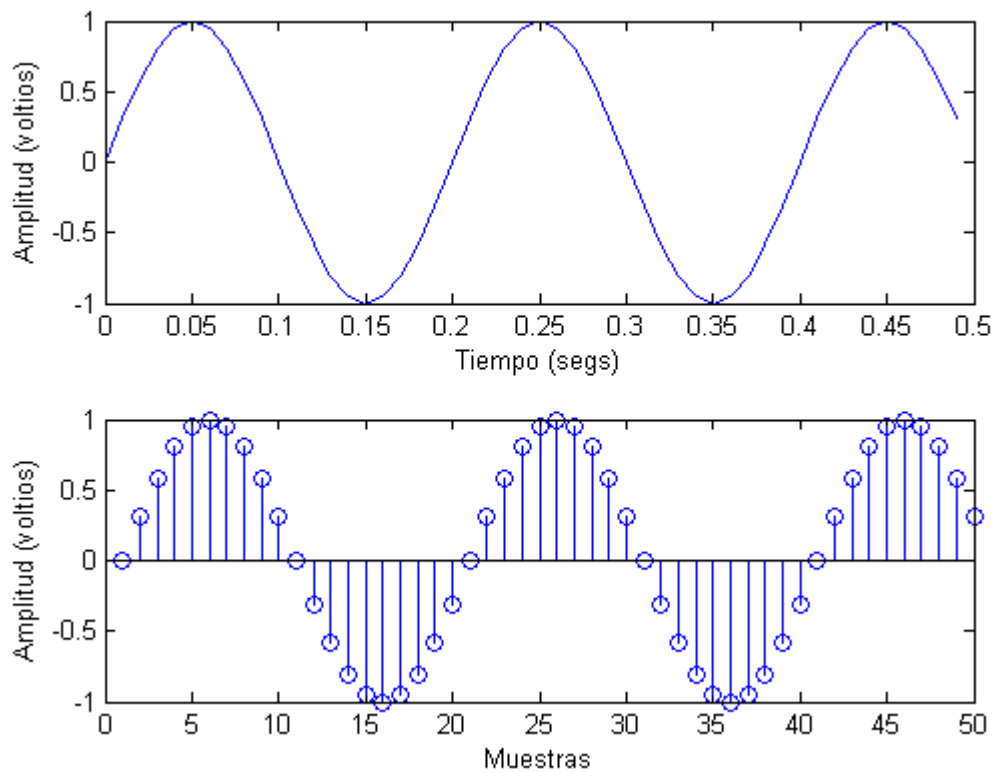
Una señal es una representación de una variable, física o numérica, que fluctúa en el tiempo. Una señal discreta será una señal que está definida dentro en un tiempo discreto, es decir, en intervalos de tiempo discontinuos. Para introducir este concepto, en la Figura 15, se ha ilustrado la fluctuación del valor del dólar durante el último año en el mercado colombiano.



**Figura 15** Fluctuación del dólar desde Mayo del año 2007 hasta el mes de Abril del 2008. Las muestras corresponden al último dato disponible para cada mes. [Portafolio.com.co](http://Portafolio.com.co).

En este caso se ha hecho una muestra por cada mes, lo cual nos indica que los datos están dados en tiempos discretos y no es posible tener acceso a la información que se encuentre fuera de estas muestras. Por ejemplo, si quisiéramos saber cómo fluctuó el dólar diariamente en el mes de mayo a partir de este gráfico, no podríamos ya que la información que tenemos disponible sería insuficiente y sólo podríamos saber cuál fue el valor puntual del dólar el último día del mes.

Otro ejemplo se muestra en la Figura 16 donde se relaciona una onda senoidal con una frecuencia de 5Hz (como podría ser el desplazamiento de una máquina rotativa) con una versión muestreada de si misma. Nótese que en la señal de tiempo continuo, la variable independiente está dada en segundos y en la de tiempo discreto está dada en muestras, esto se ha hecho a propósito para acentuar que la de tiempo discreto ha sido muestreada de la versión de tiempo continuo, pero así mismo podríamos otorgarle valores temporales correspondientes al momento preciso en que la muestra fue adquirida.



**Figura 16** Diferencia entre una señal de tiempo continuo (arriba) y una de tiempo discreto (abajo).

Basta observar la Figura 16 para darse cuenta que la primera muestra corresponde a la señal continua en cero segundos, la décima muestra corresponde al valor que adquirió esta a los 0.1 segundos y así consecutivamente. También podemos observar que en un periodo de 0.2 segundos se extrajeron 20 muestras de la señal continua, con lo que

podemos concluir que la discreta fue muestreada a una tasa de 100 muestras por segundos (100S/s).

### **2.1.3.3 Sistemas Discretos Y Sus Propiedades**

Un sistema, es un proceso físico o matemático que relaciona dos señales, una señal de entrada y una señal de salida. Un sistema discreto, no es más que un sistema cuyas señales de entrada y salida son señales discretas. Los sistemas discretos poseen algunas propiedades que nos ayudan a categorizarlos y entender sus diferentes naturalezas.

#### ***2.1.3.3.1 Sistemas Con Y Sin Memoria***

Un sistema carece de memoria si la salida en cualquier instante de tiempo depende exclusivamente de su entrada en ese mismo instante. Por ejemplo, un amplificador de audio ideal es un sistema sin memoria, ya que su salida es independiente de la historia de la entrada y solo depende del valor actual de la entrada, por lo que no presenta ningún retardo en la señal de entrada a su salida.

$$y[n] = \beta x[n], \quad \beta > 1 \quad (0.8)$$

La ecuación (0.8) describe el amplificador ideal de audio, donde  $\beta$  es el factor de amplificación. Si  $n = n - 1$ , su salida será  $y[n - 1] = \beta x[n - 1]$ . Con lo que demostramos que el sistema carece de memoria ya que su salida solamente depende del valor en el mismo instante a la entrada.

Ahora, si le aplicamos un retraso de  $k$  muestras a la señal de entrada al amplificador y describimos todo el sistema con una sola ecuación, obtendremos la relación (0.9).

$$y[n] = \beta x[n - k] \quad (0.9)$$

Cuando  $n = 0$ ,  $y[0] = \beta x[-k]$  y así para cualquier valor posible de  $n$ , con lo que demostramos que el sistema posee memoria, ya que no depende solamente del valor actual de  $x$ . Esto implica que el sistema debe almacenar al menos  $k$  valores en una memoria para su procesamiento posterior.

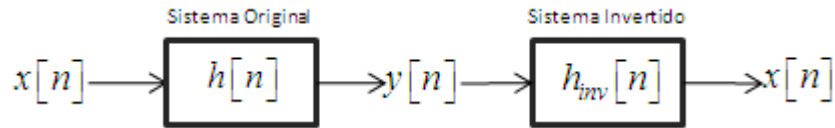
Este ejemplo nos introduce el concepto de interconexión de sistemas, el cuál trataremos a fondo en la sección 2.1.3.4, Sistemas Discretos LTI (Lineales, Invariantes en el Tiempo).

#### **2.1.3.3.2 Invertibilidad Y Sistemas Inversos**

Esta propiedad es de suma importancia en el entendimiento de la deconvolución. Un sistema es invertible si a partir de su salida podemos determinar su entrada. Siguiendo con el ejemplo del amplificador, si vinculamos su salida en serie con su sistema inverso, descrito en la ecuación (0.10), podríamos llegar a  $x[n]$  a partir de  $y[n]$ .

$$x[n] = \frac{y[n]}{\beta}, \quad \beta > 1 \quad (0.10)$$

En este caso, la entrada del sistema inverso sería la salida del amplificador y su salida correspondería a la entrada del amplificador. Este sistema inverso, en términos prácticos, no es más que un atenuador. En la figura Figura 17 se muestra la interconexión de los dos sistemas.



**Figura 17** Invertibilidad de un sistema

### 2.1.3.3.3 Causalidad

Un sistema se considera causal si su salida depende de valores presentes o pasados de su entrada. El amplificador de audio, es un sistema causal, ya que su salida no anticipa valores de su entrada. Incluso cuando vinculamos el sistema a un retraso (ver ecuación (0.9)), seguía siendo causal porque solo dependía de valores pasados de su entrada. Una observación pertinente de estos dos casos es que todo sistema sin memoria es causal pero un sistema con memoria puede o no ser causal, y esto dependerá de cada sistema independiente.

Si quisiéramos saber el valor RMS de la salida del amplificador podríamos adicionar en serie un sistema de promediado RMS como el descrito en la ecuación (0.11). Este sistema de promediado posee memoria y es no causal, ya que anticipa los valores de  $y[n]$  para hacer el cálculo de su salida.

$$y_{RMS}[n] = \sqrt{\frac{1}{N} \sum_{k=0}^{N-1} y[n+k]^2} \quad (0.11)$$

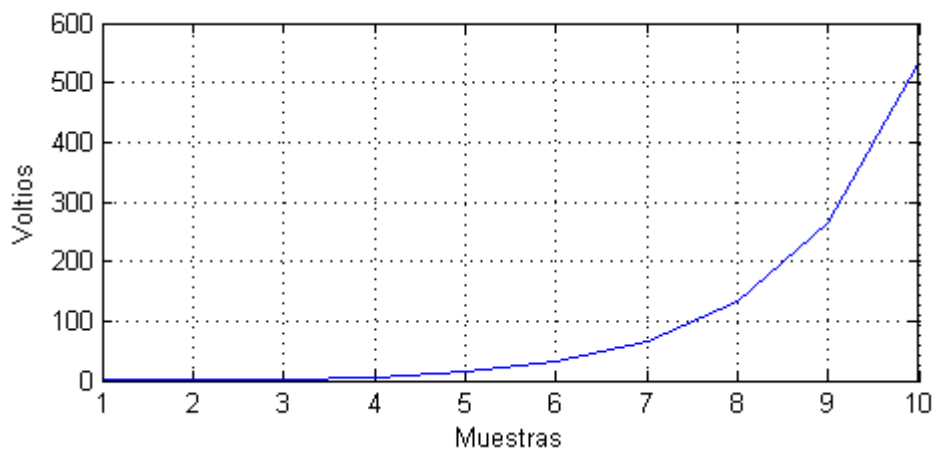
### 2.1.3.3.4 Estabilidad

Digamos que la entrada de nuestro amplificador es una señal senoidal de 1 voltio de amplitud pico. Sin importar cuánto tiempo esta sea la entrada del amplificador, su salida nunca será mayor de  $\beta x[n]$ , por lo que lo consideramos como un sistema estable. Ahora, si retroalimentamos la entrada del amplificador con su salida, como descrito en

la ecuación (0.12), su salida crecería exponencialmente sin límite, es decir que diverge o tiende a infinito.

$$y[n] = \beta(x[n] + y[n-1]) \quad (0.12)$$

La Figura 18 muestra la salida del amplificador retroalimentado y es visible que con tan solo 10 muestras para un  $\beta = 2$ , ya el sistema ha adquirido valores muy altos que seguirán creciendo indefinidamente o en el caso práctico, hasta que el amplificador resulte dañado.



**Figura 18** Curva de salida de un amplificador retroalimentado con factor de amplificación  $\beta = 2$ .

En definitiva, un sistema es estable si para una entrada limitada, su salida también está limitada o converge a un valor finito.

#### **2.1.3.3.5 Invariancia En El Tiempo**

La invariancia en el tiempo de un sistema se da cuando un desplazamiento temporal en su entrada provoca el mismo desplazamiento en su salida. Consideremos las señales de las ecuaciones (0.13) y (0.14).

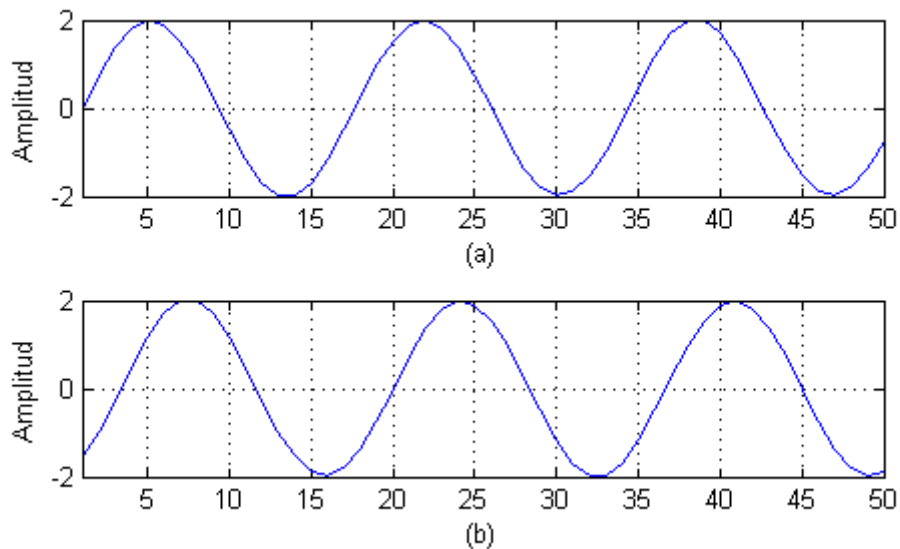
$$x_1[n] = \sin(2\pi fn) \quad (0.13)$$

$$x_2 = x_1[n-k] = \sin(2\pi f[n-k]) \quad (0.14)$$

La señal  $x_2$  es una versión de  $x_1$  con un retraso de  $k$  muestras. La **Error! Reference source not found.** muestra las salidas del sistema para cada una de las señales de entrada. La salida del sistema para  $x_2$  podría definirse como a continuación:

$$y_2 = \beta x_1[n-k] = \beta \sin(2\pi f[n-k]) \quad (0.15)$$

Con lo que corroboramos que el sistema es invariante en el tiempo, ya que un desplazamiento en su entrada se reflejará igualmente en su salida.



**Figura 19** Invariancia en el tiempo





### 2.1.3.3.6 Linealidad

La linealidad es una de las propiedades que más trataremos junto con la invariancia en el tiempo. Se dice que un sistema es lineal siempre y cuando la suma de diferentes entradas escaladas de como resultado la sumatoria de las salidas independientes de cada una de estas señales escaladas. Esta propiedad se conoce como *propiedad de superposición* y se define a continuación. En el caso de nuestro amplificador, que por el momento consideraremos que es un sistema lineal, si  $x[n] \rightarrow y[n]$ , entonces:

$$ax[n] + bx[n] \rightarrow ay[n] + by[n] \quad (0.16)$$

Como sabemos que  $y[n] = \beta x[n]$ , podemos reemplazar y llegar a la siguiente conclusión:

$$ax[n] + bx[n] \rightarrow a(\beta x[n]) + b(\beta x[n]) \quad (0.17)$$

Otra característica importante de los sistemas lineales es que si  $x[n] \rightarrow y[n]$  y a su vez  $x[n] = 0$ , entonces  $y[n] = 0$ .

### 2.1.3.4 Sistemas Discretos LTI (Lineales, Invariantes en el Tiempo)

Una de las características inherentes más explotada de los sistemas LTI es la respuesta al impulso, ya que esta nos permite conocer **toda** la información posible sobre el comportamiento de un sistema específico. Para entender qué es la respuesta al impulso

de un sistema, primero definiremos qué es un impulso en términos matemáticos. Un impulso  $\delta[n]$ , es una señal que se define de la siguiente forma:

$$\delta[n] = \begin{cases} 1 & n = 0 \\ 0 & n \neq 0 \end{cases} \quad (0.18)$$

El impulso unitario es de gran importancia en el entendimiento de los sistemas discretos por una razón, cualquier señal discreta puede ser interpretada como una serie de impulsos escalados y desplazados en el tiempo. Ver Figura 20. Estos impulsos individuales al ser superpuestos darán como resultado la señal discreta original. Esta propiedad de descomposición se conoce como *escudriñamiento*, la ecuación (0.19) representa dicha propiedad de forma general.

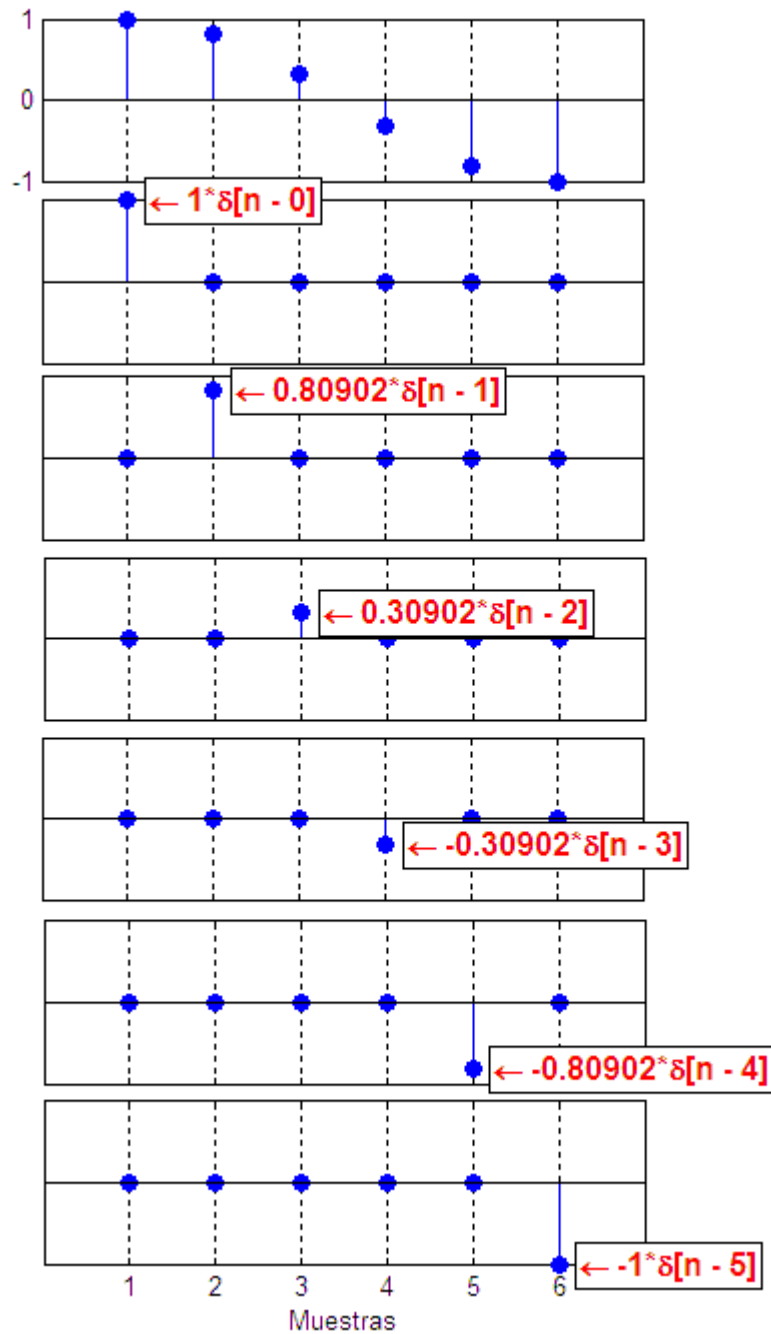
$$x[n] = \sum_{k=-\infty}^{\infty} x[k] \delta[n-k] \quad (0.19)$$

La respuesta al impulso de un sistema no es más que su salida ante un impulso unitario en su entrada.

#### **2.1.3.4.1 Sumatoria de Convolución**

La sumatoria de convolución es una característica propia de los sistemas LTI de tiempo discreto, y esta nos permite calcular la salida de un sistema LTI ante cualquier estímulo en su entrada, si se conoce a respuesta al impulso, generalmente representada como  $h[n]$ . La sumatoria de convolución se plantea a continuación:

$$y[n] = \sum_{k=-\infty}^{\infty} x[k]h[n-k] \quad (0.20)$$



**Figura 20** Descomposición de una señal discreta (arriba) en impulsos unitarios.

Hay que entender que la convolución es por si misma una operación matemática, tal como una suma relaciona la adición de dos números, la convolución relaciona tres elementos entre si, la entrada, la respuesta al impulso y la salida de un sistema y se representa con una estrella o asterisco como en la siguiente ecuación.

$$y[n] = x[n] * h[n] \quad (0.21)$$

En tiempo discreto, la convolución es equivalente a una multiplicación polinómica y se puede expresar por medio de la siguiente ecuación:

$$P(x) \cdot Q(x) = \sum_{i=0}^n \sum_{j=0}^m a_i b_j x^{i+j} \quad (0.22)$$

En esta representación,  $n$  y  $m$  son los órdenes de los polinomios respectivamente. Desde esta nueva perspectiva, podemos apreciar que el orden del polinomio resultante será  $n + m$  ya que estos son los límites de los contadores.

Esto nos lleva a un nuevo concepto, las señales en tiempo discreto pueden verse como polinomios de grado  $N-1$ , donde  $N$  es el número de muestras de la señal, si los valores de la señal y se asumen como los coeficientes de tal polinomio. Por esto, una señal que contenga 101 muestras se puede ver como un polinomio de grado 100, ya que un polinomio de grado 100 contiene 101 elementos o monomios y es por esto mismo que la convolución entre dos señales dará como resultado una señal de  $n + m - 1$  muestras, donde  $n$  y  $m$  son los tamaños respectivos de ambas señales.

### 2.1.3.5 Sistemas Físicos Reales

Casi ningún sistema físico real es LTI. En el caso del amplificador, su salida se verá afectada por el ruido eléctrico generado en su interior, su factor de amplificación variará dependiendo de la temperatura de los componentes internos y su salida se verá afectada

con un cierto porcentaje de distorsión armónica si se satura la entrada. Estos no son los únicos factores que hacen que dicho amplificador en la vida real no sea LTI, pero quizá sí los más importantes.

Igualmente pasa con un recinto acústico, cuya linealidad e invariancia en el tiempo dependerán de muchos elementos como el ruido propio del sistema eléctrico de medición, las variaciones de temperatura (ya que estas afectan la velocidad del sonido en el aire) durante la medición, y el ruido acústico de fondo, por nombrar algunos. Sin embargo, bajo las condiciones apropiadas, estos sistemas pueden ser aproximados a un modelo LTI. Por ejemplo, mejorar la relación señal-ruido hace que el ruido eléctrico o acústico pase a tomar menor importancia. También se puede controlar que las variables del sistema sean estables (i.e. temperatura, humedad, presión, etc.) y así.

En el caso de la convolución y en nuestro caso específico de la deconvolución, el éxito de dicho proceso, dependerá directamente de cuanto podemos aproximar el sistema en cuestión a un sistema LTI.

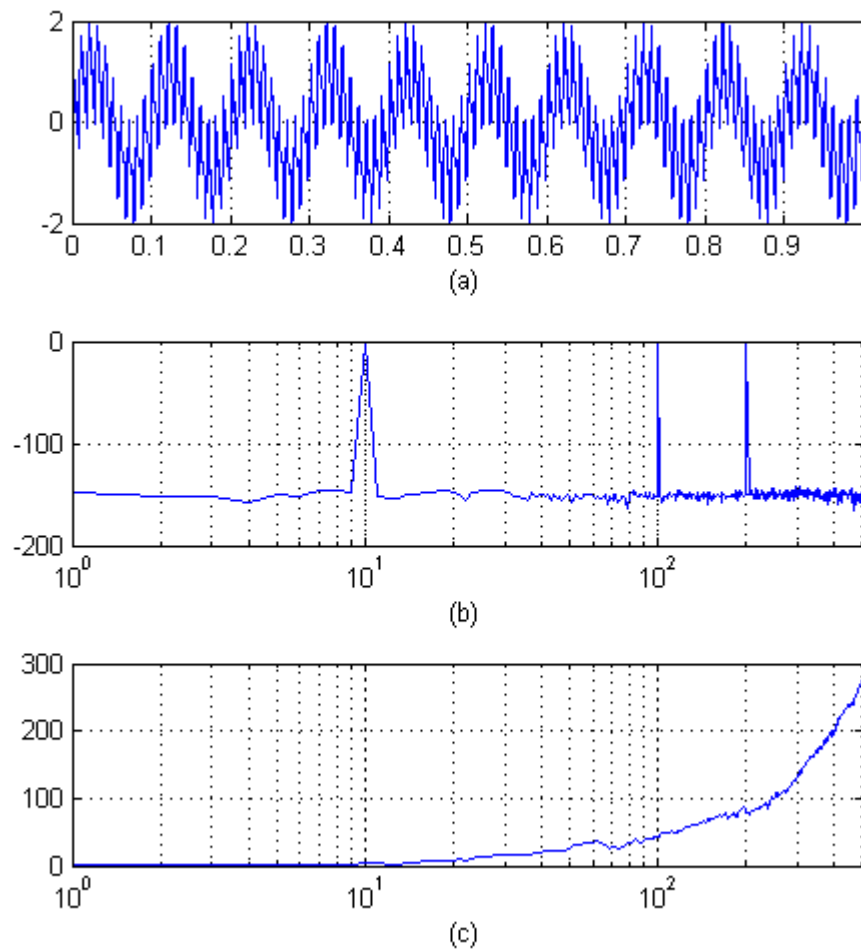
#### **2.1.3.6 Análisis Discreto De Fourier**

El análisis de Fourier parte de la base de que cualquier señal periódica puede ser descompuesta en una serie de señales senoidales de diferentes frecuencias, las cuales pueden ser recombinadas para llegar a la señal original. Para señales discretas, esto se logra a través de la *Transformada Discreta de Fourier*, que de ahora en adelante reconoceremos como DFT por sus siglas en inglés (i.e. Discrete Fourier Transform).

La ecuación para hallar la DFT de una señal es la siguiente:

$$X[k] = \sum_{n=0}^N x[n] e^{-jk(2\pi/N)n} \quad (0.23)$$

Donde  $N$  es el número de puntos a calcular,  $k$  es el contador de los componentes espectrales y  $X[k]$  es la transformada.

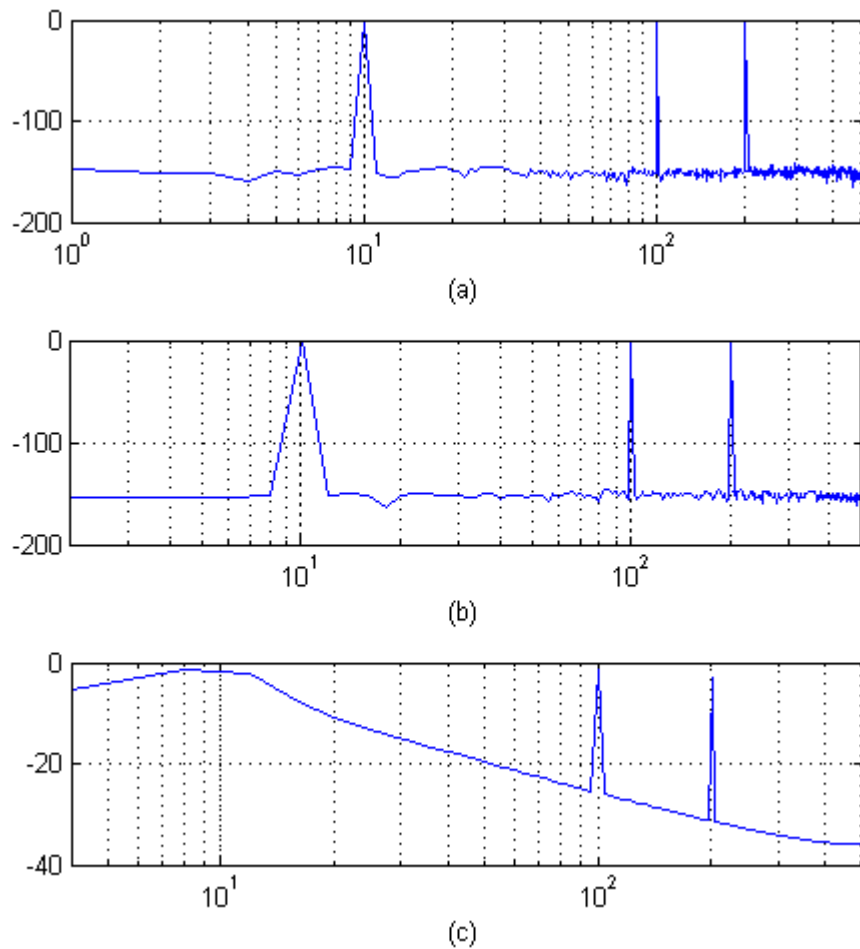


**Figura 21** Análisis de Fourier para una señal discreta. (a) Es la serie temporal, (b) Magnitud de la DFT y (c) la fase de la DFT.

En la Figura 21 se puede apreciar una señal de tiempo discreto y su DFT. El resultado de la DFT es complejo independiente de si la entrada lo es o no. En este caso, se muestra el resultado en forma polar ya que es más simple de visualizar. La señal analizada presenta varios componentes espectrales que no son fáciles de identificar en una representación original, pero si observamos la magnitud de la DFT, es claramente visible que esta señal tiene fuertes componentes senoidales en 10, 100 y 200 Hz. Esto hubiera sido muy difícil de identificar en el dominio temporal, pero es evidente en el dominio espectral. Además, tenemos información sobre la fase de esta señal para cada uno de los componentes espectrales.

La DFT no es más que otra forma de analizar una misma información y en ningún caso se gana información adicional transformando una señal por medio de la DFT. Sin embargo, es posible perder información si el tamaño de la DFT es menor que el número de muestras a transformar. En la Figura 22 se muestra tal diferencia, y es notable como la resolución espectral (i.e. espaciamiento entre componentes espectrales) disminuye cada vez que se hace menor el número de puntos de la transformada.

En el último caso, donde la transformada se hace con 250 puntos, es notable la pérdida de información valiosa en el proceso, y esta información no es recuperable a partir de la transformada inversa, es decir, la transformada inversa (i.e. IDFT de dominio espectral a dominio temporal) no dará la señal original sino una versión con menos resolución temporal.



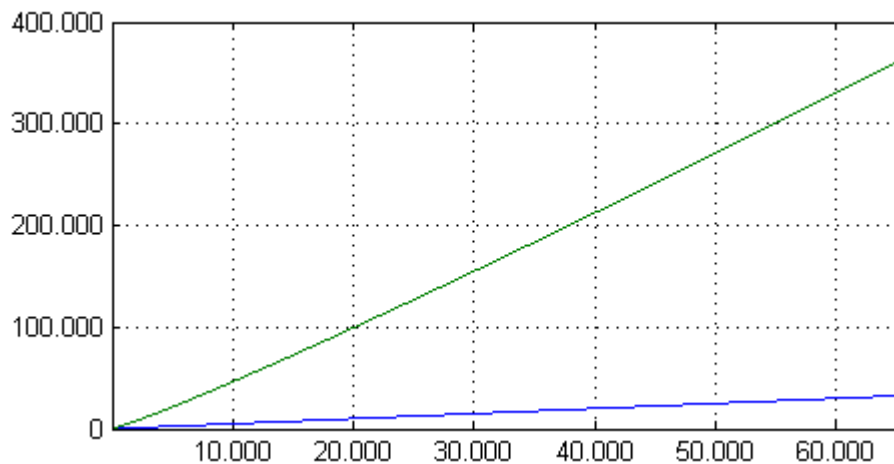
**Figura 22** Resolución espectral. (a) 1000 puntos, (b) 500 puntos y (c) 250 puntos.

### 2.1.3.6.1 FFT: Transformada Rápida De Fourier

La FFT o transformada rápida de Fourier (i.e. Fast Fourier Transform) es un algoritmo para computar la DFT de forma más eficiente. El primer algoritmo de FFT fue el de Cooley-Tukey, nombrado así por sus autores y fue publicado en 1965, pero fue solo hasta hace poco que cobró vida, ya que en ese entonces no había computadoras capaces de procesar tal algoritmo en tiempos aceptables. Podría decirse que hoy en día ha reemplazado la computación tradicional de la DFT.



Para calcular la DFT según la ecuación (0.23), tocaría hacer  $O(N^2)$ <sup>1</sup> operaciones aritméticas, mientras que para calcular la DFT mediante la FFT, solo se tendrían que calcular  $O(N \log N)$  operaciones.



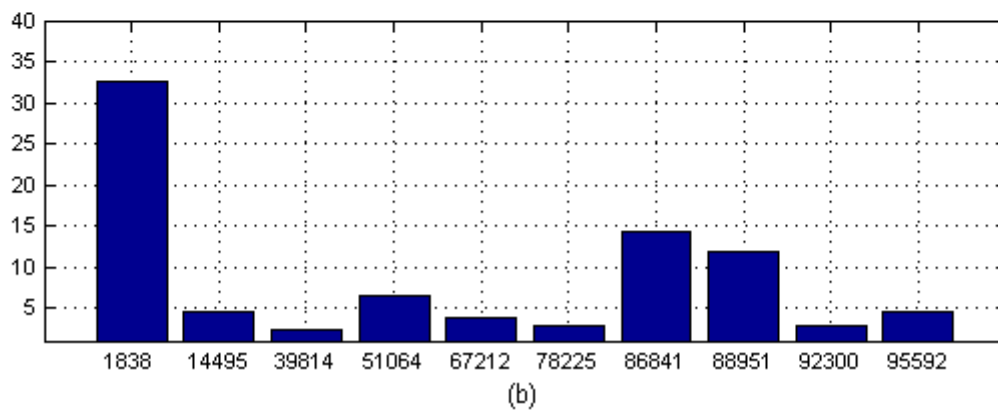
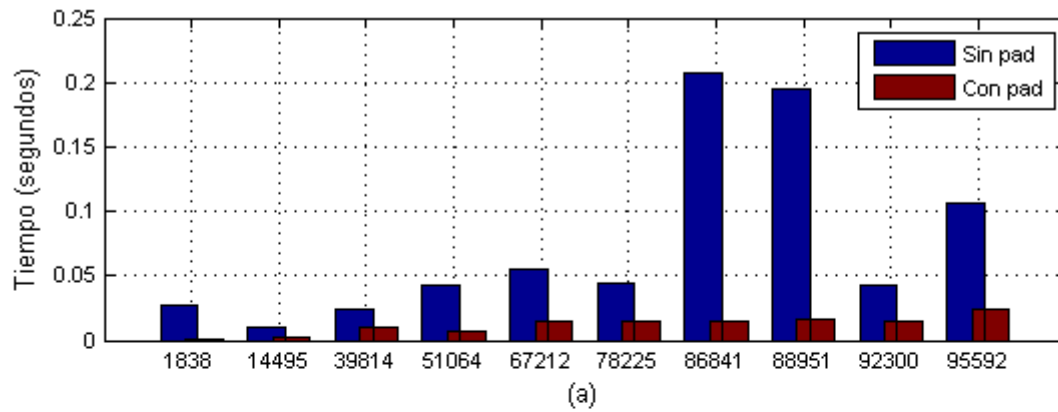
**Figura 23** Tamaño de la DFT vs. Número de operaciones aritméticas. DFT tradicional (verde) y FFT (azul).

En la Figura 23 podemos observar la diferencia computacional entre el método tradicional de computar la DFT y el método FFT. Salta a la vista que la FFT se vuelve cada vez más eficiente si consideramos el número de operaciones necesarias para un determinado tamaño de DFT, mientras que el método tradicional para calcular la DFT crece de forma lineal, es decir, el número de operaciones es directamente proporcional al número de puntos de la transformada.

Debido a la naturaleza del algoritmo de la FFT, este es mucho más eficiente si el número de la transformada es potencia de dos (i.e.  $2^n$ ). En la Figura 24 se muestra cómo la FFT es más eficiente en tiempos de procesamiento si el tamaño de la transformada es potencia de dos. Los datos de la figura son el resultado de realizar la FFT en secuencias de números aleatorios y de tamaños aleatorios entre 1024 y 131072. En la parte (a) de la figura, se pueden ver dos series de datos: una azul, correspondiente

<sup>1</sup> Esta notación hace referencia al costo computacional de un algoritmo. Ver referencia.

a la FFT de la secuencia aleatoria; una roja, correspondiente a la FFT de la secuencia aleatoria con ceros agregados al comienzo y fin de la secuencia para hacer que su tamaño sea potencia de dos.



**Figura 24** Eficiencia del FFT dependiendo del tamaño de la secuencia.

Incluso cuando el tamaño de la DFT es mayor debido a los ceros adicionales, la FFT resulta más eficiente para secuencias cuyos tamaños son potencias de dos. Por ejemplo, en el caso de la secuencia de 95592 muestras, si se ejecuta la FFT con la misma cantidad de puntos, el tiempo de procesamiento es poco más de 0.1 segundos. Pero si la secuencia se rellena con ceros al comienzo y al final para hacer que su tamaño sea de 131072, resultando en una secuencia con 35480 muestras adicionales, la FFT toma alrededor de 0.025 segundos. En la parte (b) de esta figura, se muestra la relación entre la FFT con pad (i.e. ceros para hacer su tamaño potencia de dos) y la FFT sin pad y para el mismo caso, vemos que la FFT con pad es casi 5 veces más rápida que la FFT sin pad.

## **2.2 MARCO TEÓRICO**

### **2.2.1 HISTORIA Y DECONVOLUCIÓN**

#### **2.2.1.1 Introducción**

Para poder entender el alcance y metas de este proyecto, es importante que retomemos un poco sus usos a través de la historia, sus usos presentes, y finalmente, entendamos el posible uso del algoritmo de deconvolución planteado en este proyecto en un futuro.

Como hemos mencionado anteriormente, la deconvolución es un proceso matemático que hasta hace muy poco era exclusivamente empleado de forma teórica y no tenía aplicación práctica en sistemas reales debido al gran poder de procesamiento requerido, pero los computadores cada vez son más poderosos, permitiéndonos acceder a nuevas técnicas y tecnología con cada día que pasa y por supuesto, la deconvolución no es la excepción.

Desde hace ya un par de décadas que la deconvolución adquirió un papel importante en el post-tratamiento de señales, pero debido al incremento de capacidad de procesamiento de los sistemas de hoy, ha sido posible la deconvolución en tiempo real, lo que marca un hito en la historia.

#### **2.2.1.2 Audio Y Acústica**

En el campo del audio y acústica, la deconvolución ha hecho posible procesos de filtrado que antes eran imposibles.

En acústica, se ha empleado principalmente para obtener la respuesta al impulso de un sistema cuando este es estimulado por una señal no impulsiva. Por ejemplo, varios estímulos no impulsivos, como ruido blanco, ruido rosa, MLS o LSS son utilizados comúnmente en la medición de la respuesta al impulso de un recinto.

Imaginemos el siguiente esquema: conectamos la salida de audio de un computador (podría ser a través de una interface externa) a una fuente acústica omnidireccional; Un micrófono de medición (i.e. omnidireccional, respuesta en frecuencia plana en todo el espectro audible, etc.) está conectado a la entrada de audio del computador. Si generamos constantemente ruido rosa y captamos con el micrófono el ruido generado dentro del recinto, podríamos deconvolucionar el ruido captado con el ruido generado para hallar la respuesta al impulso del recinto ya que conocemos las otras dos variables: su entrada y su salida.

### **2.2.1.3 Imágenes Y Video**

En el procesamiento de imágenes, la deconvolución ha entrado hasta el momento el mismo uso que en los otros campos, filtrar. Es muy común ver algoritmos para restaurar una imagen que ha sido degradada de alguna forma. Por ejemplo, en los microscopios ópticos modernos se ha comenzado a utilizar la deconvolución para revertir los efectos negativos, como bordes fuera de foco o contraste no adecuado para una buena visualización, debidos a la apertura limitada del objetivo.

Es común encontrar monitores médicos con capacidad de deconvolucionar la imagen adquirida para mejorar su visualización, lo cual mejora la precisión del diagnóstico. Para lograr una deconvolución óptima, se someten los objetivos a condiciones lumínicas especiales, como iluminación fluorescente u otras.

#### **2.2.1.4 Otros**

En general, la deconvolución ha demostrado ser muy útil para filtrar alteraciones no deseadas en una señal que de otro modo requerirían diseños de filtros muy complejos y muy difíciles de implementar.

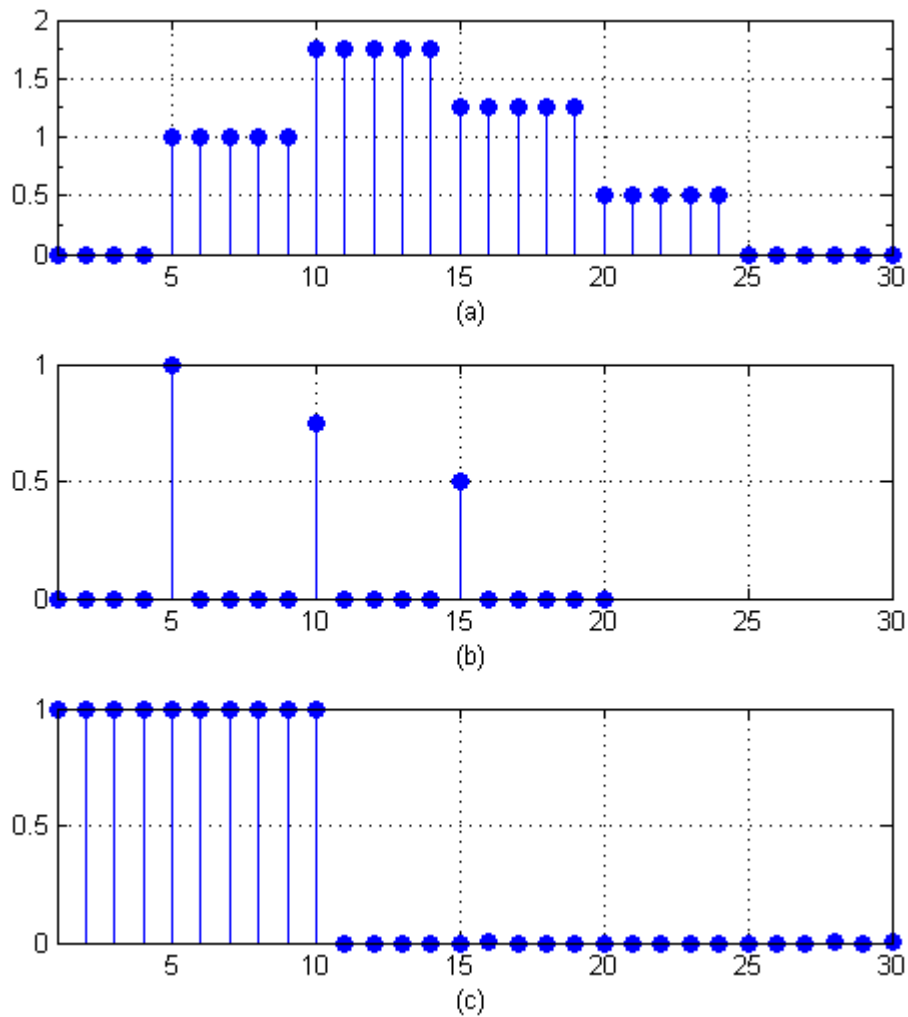
### **2.2.2 DECONVOLUCIÓN DISCRETA**

#### **2.2.2.1 Introducción**

Como se habló anteriormente, la deconvolución para sistemas reales, solo ha sido posible desde hace poco tiempo. Esto es debido a la complejidad del procesamiento de señales involucrado que de forma análoga es muy difícil de conseguir eficientemente, por lo cual este proceso siempre se ha logrado de forma digital, sin importar el fin del mismo y es por eso que solo trataremos a profundidad la deconvolución para señales y sistemas discretos.

#### **2.2.2.2 Sistemas Invertibles Y Teorema De Deconvolución**

Según la teoría de los sistemas invertibles, si un sistema tiene un sistema inverso, es posible obtener mediante su salida, su entrada. Esto es el principio básico de la deconvolución. En Figura 25 se muestra el resultado de esta operación para un sistema de retrasos bastante simple.



**Figura 25** Señal alterada con retrasos (a), respuesta al impulso del sistema de retrasos (b) y la señal deconvolucionada (c).

En este caso, la deconvolución es comprobable a simple vista. Si miramos la señal de salida del sistema de retrasos y la comparamos con la respuesta al impulso de este, se hace obvio que la señal de entrada debió ser la señal de la Figura 25 (c). Analicemos la señal de entrada por partes. Primero podemos ver que de la primera muestra a la quinta, la señal contiene solo ceros y su respuesta al impulso también, por lo que asumimos que la primera muestra de su entrada fue un impulso. Luego, vemos que de la quinta muestra a la novena, la amplitud de la señal retrasada es 1, por lo que podemos asegurar que el primer impulso de la entrada persistió cuando menos durante cinco muestras. Ahora viene la parte no tan obvia, la señal (a) muestra un aumento en su amplitud de 1 a 1.75 de la décima muestra a la catorceava, pero dada la respuesta al

impulso, podemos ver que la primera repetición se hace en la décima muestra y atenúa el impulso en 0.75. A raíz de esto, podemos concluir que la señal de entrada persistió por lo menos cinco muestras adicionales con una amplitud de uno, ya que se superpone con el segundo retraso del sistema  $(1 + 0.75)$ .

De esta forma podemos seguir analizando la señal de salida del sistema y compararla con su respuesta al impulso para determinar su entrada. Este ejemplo resulta ilustrativo, pero en los sistemas físicos reales las respuestas al impulso y las señal de entrada son mucho más complejas por lo que resultaría difícil determinar la deconvolución de esta misma forma gráfica.

### 2.2.2.3 Métodos De Aproximación A La Deconvolución

Existen dos métodos para efectuar la deconvolución: división polinómica y deconvolución espectral.

#### 2.2.2.3.1 *División Polinómica (Dominio temporal)*

La convolución en tiempo discreto es una multiplicación polinómica, y como sería de esperarse, la deconvolución en tiempo discreto es una división polinómica. A continuación se elabora una división entres dos polinomios:

$$\begin{array}{r}
 4x^4 + 3x^3 + 2x^2 + 1 \quad | \quad x^3 - 2x^2 - 1 \\
 \underline{-4x^4 + 8x^3 + 4x} \quad 4x + 11 \\
 11x^3 + 6x^2 + 1 \\
 \underline{-11x^3 + 22x^2 + 11} \\
 28x^2 + 12
 \end{array} \tag{0.24}$$

La división de polinomios se realiza de forma similar a la división aritmética. El primer paso es dividir el monomio de mayor grado del dividendo sobre el monomio de mayor grado del divisor. Luego, se multiplica el resultado de dicha operación por todos los elementos del divisor para obtener un nuevo polinomio, el cual se invierte de signo para ser sumado con el dividendo. Así logramos reducir el grado del dividendo en un grado, y continuamos haciendo esto hasta obtener el grado que se desea.

En este ejemplo es notable que el dividendo es un polinomio de cuarto grado, el divisor es un polinomio de tercer grado y el grado del cociente es la resta de los grados del dividendo y el divisor (i.e. primer grado en este caso), por lo que se hace obvio que el dividendo siempre tendrá que ser de mayor grado que el divisor para que la división de un resultado válido. El módulo podrá ser máximo un grado menor que el cociente.

#### ***2.2.2.3.2 FFT: Transformada Rápida De Fourier (Dominio espectral)***

Una de las propiedades de la DFT es que la multiplicación de dos series de coeficientes espectrales es igual a la convolución de las series temporales correspondientes y por ende, la deconvolución en el dominio frecuencial equivale a la división de los dos espectros. A diferencia de la división polinómica, la deconvolución implementada por FFT es mucho más rápida, ya que se requieren  $O(N \log N)$  en vez de  $O(N^2)$  (la misma cantidad de operación requeridas para calcular la DFT por la sumatoria discreta de Fourier), a tal punto que podría implementarse bajo procesamiento recursivo un sistema con una latencia mínima, es decir, se podría obtener la deconvolución en tiempo real. El punto en contra de esta aproximación a la deconvolución es la dificultad programática en comparación a la división polinómica, ya que la operación pasa de ser una sumatoria en el dominio temporal a una transformación al dominio espectral para luego efectuar la división y a partir de este resultado, volver a transformar la señal al dominio temporal.



### **3 METODOLOGÍA**

#### ***3.1 ENFOQUE DE LA INVESTIGACIÓN***

Esta investigación se realizará bajo un enfoque empírico-racional, ya que se tendrán en cuenta las observaciones experimentales de la implementación del algoritmo ADAO, así como el desarrollo lógico y racional del mismo, que permita hallar soluciones prácticas y confiables a los problemas que se presenten (i.e. excesos de tiempos de procesamiento).

#### ***3.2 LÍNEA DE INVESTIGACIÓN DE USB / SUB-LÍNEA DE FACULTAD / CAMPO TEMÁTICO DEL PROGRAMA LINEA DE INVESTIGACION***

Se ha definido como campo temático el *Campo de diseño de sistemas de sonido* ya que el ADOA en sí es un filtrado a un sistema de acústico o de audio. La sub-línea en la cual encaja el proyecto es en la línea de *Procesamiento de señales* ya que como cualquier otro filtro, independientemente de sus características (i.e. de tiempo discreto o continuo, FIR o IIR, etc.), es un procesamiento de señal basado en un objetivo puntual, en este caso, limpiar o filtrar las alteraciones causadas por un sistema particular. La línea de investigación de la USB escogida fue la de *Tecnologías actuales y sociedad*, ya que se pretende establecer el uso de una tecnología que sólo ha sido práctica hasta hace poco en un problema que afecta a la sociedad ya que este algoritmo de deconvolución podría tener aplicaciones incluso en acústica forense.

### **3.3 TÉCNICAS DE RECOLECCIÓN DE INFORMACIÓN**

La técnica que se empleará para la recolección de datos será totalmente empírica, es decir, esta se obtendrá a partir de mediciones de la respuesta al impulso y relación entrada-salida (función de transferencia) de los distintos sistemas acústicos por analizar. Se analizarán, la mayor cantidad de sistemas posibles, contemplando recintos acústicos, equipos electrónicos y demás sistemas que puedan presentar un aporte a esta investigación.

### **3.4 HIPÓTESIS**

“Por medio de procesamiento digital de señales se puede lograr un algoritmo confiable de deconvolución para sistemas acústicos reales”

“Por medio de procesamiento digital de señales **no** se puede lograr un algoritmo confiable de deconvolución para sistemas acústicos reales”

### **3.5 VARIABLES**

#### **3.5.1 VARIABLES INDEPENDIENTES**

- Propiedades no lineales de los sistemas
- Equipos de medición
- Estándares de medición

### **3.5.2 VARIABLES DEPENDIENTES**

- Tiempo de procesamiento del algoritmo
- Precisión del algoritmo
- Lenguaje de programación
- Sistemas a medir

## **4 DESARROLLO INGENIERÍL**

### ***4.1 METODOLOGÍA DE DESARROLLO DEL ALGORITMO***

#### **4.1.1 INTRODUCCIÓN**

La estructura o metodología del desarrollo del algoritmo es tan importante como el resultado de este. Por eso, se ha escogido un esquema de “diseño de software en cascada”, donde cada etapa depende de forma incondicional de la finalización de la etapa inmediatamente anterior. Esto nos permite tener un mayor control sobre el flujo de trabajo en el algoritmo, ya que cada etapa de diseño está separada de las otras, lo cual representa una ventaja estructural.

#### **4.1.2 ANÁLISIS DE REQUISITOS**

Los requisitos del software se han dividido en dos partes, requisitos de la GUI (i.e. Interfaz Gráfica de Usuario) y requisitos de procesamiento de datos:

- GUI
  - La GUI debe contar con un objeto gráfico que permita cargar el kernel o respuesta al impulso a procesar.
  - Un objeto gráfico debe correr algoritmo y parar algoritmo a petición del usuario.
  - La GUI contará con dos objetos gráficos que permitan visualizar el kernel tanto en tiempo como en el dominio espectral.
  - La información (i.e. tamaño en muestras, duración en segundos, etc.) de las señales involucradas en la deconvolución: kernel, entrada y salida.
  - Debe contar con un elemento que permita cerrar la interfaz y borrar todos los datos relacionados con esta o con el procesamiento de datos.

- También se debe contar con un panel de propiedades donde se le permita al usuario cuadrar los parámetros de procesamiento como frecuencia de muestreo, resolución, y tamaño de bloque de procesamiento.
- Adquisición y procesamiento de datos
  - El software debe adquirir los datos provenientes de una entrada de audio predeterminada.
  - Los datos adquiridos a través de la interfaz de audio deben ser procesados con el kernel cargado en el GUI para lograr la deconvolución.
  - Los datos resultantes de la deconvolución deben ser sacados por una salida análoga en la interfaz de audio.

### 4.1.3 DISEÑO

El diseño del software se hará de forma consecuente con cada etapa de la siguiente forma:

1. Planteamiento y compilación de la GUI: en esta etapa se diseña la GUI que permitirá al usuario interactuar con el algoritmo de deconvolución de forma gráfica.
2. Inicialización de GUI y estructura de datos: en esta etapa se llamará a la figura del GUI y la estructura de los accesos a los objetos gráficos. También se inicializará toda la estructura de datos. A continuación se plantean los esquemas de ambas estructuras:

- Data
  - ai: en esta variable se guardará un objeto de tipo ‘analoginput’ correspondiente a la entrada de audio predeterminada.

- aiChannels: aquí se definirá el canal del objeto ai a utilizar.
- ao: en esta variable se guardará un objeto de tipo 'analogoutput' correspondiente a la salida de audio predeterminada.
- aoChannels: aquí se definirá el canal del objeto ao a utilizar.
- iSegment: contador de segmentos adquiridos y procesados.
- meanProcessing: tiempo promedio de procesamiento de un segmento.
- pow: valor de la potencia de dos que define el tamaño final de la deconvolución.
- nyquist: frecuencia de nyquist correspondiente al tamaño DFT escogido.
- freq: vector de frecuencias correspondientes a la DFT.
- kernel
  - signal: vector de muestras del kernel.
  - ln: tamaño del vector kernel.signal
  - time: vector tiempo del kernel.
  - toPad: cantidad de ceros a añadir al segmento de entrada para su procesamiento.
  - fft: vector complejo recipiente de la DFT del kernel.signal.
  - fftLn: tamaño del vector kernel.fft.
- input
  - toPad: cantidad de ceros a añadir al segmento de entrada para su procesamiento.
  - time: vector tiempo de cada segmento de entrada.
  - segment: vector de entrada de cada segmento de procesamiento.
  - total: vector de entrada total.
  - fft: vector complejo recipiente de la DFT del input.signal.
  - fftLn: tamaño del vector input.fft.
- output
  - time: vector tiempo de cada segmento de salida.
  - overlap: vector a acumular el segmento a superponer con el próximo segmento de entrada.

- segment: vector de salida de cada segmento de procesamiento
- total: vector de salida total, es decir, la suma de todos los vectores segmentos de salida más el último segmento a superponer.
- totalLn: tamaño del vector output.total.
- totalTime: vector tiempo correspondiente a output.total.

#### ○ Handles

- mainFig: figura principal.
- runstop: botón estilo 'toggle' para correr y parar el procesamiento del algoritmo.
- help: menú 'Help' para futura implementación de una ayuda documentada.
- file: menú 'File'.
- samplerate\_pop: menú desplegable para selección de frecuencia de muestreo.
- blocksize\_pop: menú desplegable para selección del tamaño del bloque de procesamiento.
- output: axes de visualización de la salida.
- kernel\_play: botón para escuchar el kernel cargado.
- kernel\_fft: axes de visualización de la DFT del kernel.
- kernel: axes de visualización en dominio temporal del kernel.
- about: ítem del menú 'Help' para implementación futura de un 'Acerca de...' donde se incluirá información básica del software, como autor, versión y fecha de última revisión.
- content: ítem del menú 'Help' para implementación futura del contenido de la ayuda de usuario.
- close: ítem del menú 'File' para cerrar la figura principal.
- save\_deconvolved: ítem del menú 'File' para futura implementación que permita guardar la señal deconvolucionada.
- save\_unaltered: ítem del menú 'File' para futura implementación que permita guardar la señal de entrada sin alteraciones.

- `load_kernel`: ítem del menú 'File' que permite cargar el kernel deseado.
  - `bit24_radio`: radio para ajustar la resolución de captura a 24 bits.
  - `bit16_radio`: radio para ajustar la resolución de captura a 16 bits.
  - `dBrmsBar`: axes de monitoreo de niveles de entrada.
3. Definición de los Callbacks<sup>2</sup> de la GUI: en esta etapa se definen las funciones a ejecutar correspondientes a cada uno de los eventos de los objetos de la GUI. Por ejemplo, ejecutar o parar la ejecución de la deconvolución cuando el botón Run/Stop es presionado.
  4. Definición de sub-operaciones de los Callbacks o de procesamiento: acá se definen todas las funciones restantes que tengan que ver con el procesamiento (i.e. restablecer los valores para las variables iniciales cada vez que se ejecuta el algoritmo) o las sub-funciones de los Callbacks (i.e. graficar la serie temporal y su respectiva DFT cuando el kernel ha sido cargado).
  5. Definición del Callback de deconvolución: esta es la etapa más importante en cuanto al procesamiento de datos. En ella, se define la función a ejecutar cada vez que el objeto de la entrada de audio adquiere el número de muestras determinado por `blocksize`. Esta función se encarga de calcular la deconvolución de la entrada por medio de la aproximación FFT Overlap-Add.
  6. Definición de librerías adicionales: acá se implementarán tres librerías que no entran en ninguna de las categorías anteriores, entre esas, una librería para hacer promediados lineales, una para convertir a decibeles un valor de entrada y una para añadir los seros necesarios para que la entrada alcance la siguiente potencia de dos en tamaño.

---

<sup>2</sup> Los Callbacks, en programación orientada a eventos, son las funciones que se ejecutan debido a un evento específico.



## 4.1.4 CODIFICACIÓN

### 4.1.4.1 adoa.m

Esta función se encarga de llamar las funciones que inicializan la estructura Data y la estructura Handles (definida previamente en la figura de la GUI). En este proceso, abre la figura de la GUI y de estar disponible, llama a la función de monitoreo de niveles de entrada. Para ejecutar el programa desde línea de comando, es esta función la que toca llamar. Esta función carece de parámetros de entrada o salida ya que su papel es solamente invocar otras funciones.

```
function adoa
% This is the ADOA's help!!!
global Data Handles

Data = init_setup;
Handles = guihandles(openfig('adoa.fig'));
operations('init_gui');
operations('setforpeeking');
```

### 4.1.4.2 gui\_callbacks.m

Esta función define y controla los callbacks de la GUI. Aquí están incluidas todas las funciones correspondientes a los objetos gráficos como botones, menús desplegables, etc. Sus argumentos de entrada son hObject y eventdata, los cuales son indispensables para cualquier Callback dentro de MATLAB. hObject es la referencia del objeto que genera el evento y eventdata son algunas especificaciones del evento en sí, como nombre, hora en que ocurrió el evento, número de veces que ha ocurrido (en caso de un evento repetitivo), por nombrar algunos.

Luego de recibir sus argumentos de entrada, `gui_callbacks` evalúa la propiedad `Tag`<sup>3</sup> de `hObject` para luego evaluarla y definir la sub-función a ejecutar.

```
function gui_callbacks(hObject, eventdata)

global Data Handles

tag = get(hObject, 'Tag');
eval(tag);

%% Sample Rate selection
function samplerate_pop
    % Update sample rate
    str = get(Handles.samplerate_pop, 'String');
    val = get(Handles.samplerate_pop, 'Value');

    % Stop analog input to set new sample rate and start again
    stop(Data.ai);
    set(Data.ai, 'SampleRate', str2double(str{val}));
    start(Data.ai);
end

%% Bit depth selection
function bit16_radio
    % Update bit depth
    val = get(Handles.bit16_radio, 'Value');
    set(Data.ai, 'BitsPerSample', 16);
end

function bit24_radio
    % Update bit depth
    val = get(Handles.bit24_radio, 'Value');
    set(Data.ai, 'BitsPerSample', 24);
end

%% Blocksize selection
function blocksize_pop
    % Update blocksize
    str = get(Handles.blocksize_pop, 'String');
    val = get(Handles.blocksize_pop, 'Value');

    % Stop object to set new block size and start again
    stop(Data.ai);
    set(Data.ai, 'SamplesAcquiredFcnCount', str2double(str{val}));
    start(Data.ai);

    % Update FFT variables and dependants
    operations('update_fft');
end

%% Run/Stop Togglebutton Callback
function runstop
```

---

<sup>3</sup> La propiedad 'Tag' de un objeto guarda una variable 'char' que sirve para que el programador le de una identificación única a este.

```

if get(hObject, 'Value')
    operations('reset_data');
    set(hObject, 'String', 'Stop');
    trigger(Data.ai)
else
    set(hObject, 'String', 'Run');
    stop(Data.ai)
    Data.iSegment = Data.iSegment - 1;
    step = 1/Data.ai.SampleRate;

    % Initialize
    Data.input.total = zeros(1, Data.iSegment *
    Data.ai.SamplesAcquiredFcnCount);
    Data.output.total = [Data.input.total zeros(1,
    Data.input.toPad)];

    for i = 1:Data.iSegment
        Data.input.total((i -
        1)*Data.ai.SamplesAcquiredFcnCount +
        1:i*Data.ai.SamplesAcquiredFcnCount)...
            = Data.input.segment{i};
        Data.output.total((i -
        1)*Data.ai.SamplesAcquiredFcnCount +
        1:i*Data.ai.SamplesAcquiredFcnCount)...
            = Data.output.segment{i};
    end
    Data.output.total = [Data.output.total(1:end -
    Data.input.toPad) Data.output.overlap...
    (1:Data.kernel.ln - 1)];
    Data.output.totalLn = length(Data.output.total);
    Data.output.totalTime = 0:step:
    (Data.output.totalLn * step) - step;

    % Plot total output
    plot(handles.output,...
        Data.output.totalTime, Data.output.total)

    % Rearm analog input
    start(Data.ai)
end
end
%% Play kernel
function kernel_play
    sound(Data.kernel.signal, Data.ai.SampleRate,
    Data.ai.BitsPerSample)
end
%% Load Kernel Menu Callback
function load_kernel
    [file path] = uigetfile('*.wav');

    if file ~= 0
        if path ~= 0
            % Read WAVE file
            [Data.kernel.signal Data.ai.SampleRate] =
            wavread([path file]);
            Data.kernel.signal = Data.kernel.signal';
            Data.kernel.ln = length(Data.kernel.signal);

            % Kernel time definition
            step = 1/Data.ai.SampleRate;

```

```

limit = Data.kernel.ln / Data.ai.SampleRate;
Data.kernel.time = 0:step:limit - step;

% Update FFT variables and dependants
operations('update_fft');

% Plot kernel waveform
plot(Handles.kernel,...
     Data.kernel.time, Data.kernel.signal,...
     'Color', [0 0 1]);
set(Handles.kernel,...
     'XLim', [Data.kernel.time(1)
Data.kernel.time(end)])

% Plot kernel FFT
toPlot =
abs(Data.kernel.fft(2:length(Data.freq))
*(2/Data.kernel.fftLn));
toPlot = 2*unit2db(toPlot);
plot(Handles.kernel_fft,...
     Data.freq(2:end), toPlot,...
     'Color', [0 0.5 0])
set(Handles.kernel_fft,...
     'XLim', [Data.freq(2) Data.freq(end)])

% Update kernel information
operations('update_kernel_info');

% Activate Run/Stop and play Button
set([Handles.runstop Handles.kernel_play],...
     'Enable', 'on');
end
end
end
end
end

```

#### 4.1.4.3 init\_setup.m

En init\_setup.m se inicializa la estructura Data, donde se acumulan todas las variables correspondientes al procesamiento de las señales y las características de estas. También se definen los Callbacks de los objetos de adquisición y salida de datos. Esta función no tiene argumentos de entrada pero si tiene uno de salida, Data.

```

function [Data] = init_setup()
% This is the ADOA's init setup help!!!

%% Reset possible existing variables
delete(daqqfind)

```

```

clear Data

%% Analog Input Setup
% AI Object Construction
Data.ai = analoginput('winsound', 0);
Data.aiChannels = addchannel(Data.ai, 1:1);

% Sample Rate and Bit Depth Setup
set(Data.ai,...
    'StandardSampleRates'      , 'Off'          ,...
    'SampleRate'               , 48000          ,...
    'BitsPerSample'           , 16             ,...
    'SamplesPerTrigger'       , Inf           ,...
    'LoggingMode'              , 'Memory'     ,...
    'LogFileName'              , ''          ,...
    'TriggerType'              , 'Manual'    ,...
    'TriggerCondition'         , 'None'     ,...
    'TriggerRepeat'            , 0          ,...
    'Tag'                      , 'aistop'    ,...
    'StopFcn'                  , ''         ,...
    'SamplesAcquiredFcnCount'  , 2^14       ,...
    'SamplesAcquiredFcn'      , @deconvolve ,...
    'TriggerFcn', '');

%% Init analog output object
% AO Object Construction
Data.ao = analogoutput('winsound', 0);
Data.aoChannels = addchannel(Data.ao, [1 2]);

% Sample Rate and Bit Depth Setup
set(Data.ao,...
    'StandardSampleRates'      , 'Off'          ,...
    'SampleRate'               , Data.ai.SampleRate ,...
    'BitsPerSample'           , Data.ai.BitsPerSample ,...
    ,...
    'TriggerType'              , 'Immediate'    ,...
    'Tag'                      , 'aostop'       ,...
    'StopFcn'                  , ''           ,...
    'TriggerFcn', '');

%% Init Variables
% Time setup
step = 1/Data.ai.SampleRate;
timeLim = Data.ai.SamplesAcquiredFcnCount / Data.ai.SampleRate;
Data.time = 0:step:timeLim - step;

% Adoa settings

% Empty output signal
Data.output.signal = [];
Data.iSegment = 1;
Data.meanProcessing = 0;

end

```

#### 4.1.4.4 operations.m

La function de operations.m realiza una variedad de procedimientos genéricos que son necesarios para el procesamiento de datos y que podrían considerarse como sub-funciones de los callbacks definidos en gui\_callbacks.m. En este caso, su argumento de entrada es tag, y corresponde a una línea de texto la cual luego se evalúa. No tiene argumentos de salida ya que solo opera sobre variables globales.

```
function operations(tag)

global Data Handles

if isa(tag, 'function_handle')
    tag();
elseif ischar(tag)
    eval(tag);
else
    error('Tag argument should be a string or a function handle!')
end

%% Init gui components
function init_gui
    % Set sample rate popup
    str = get(handles.samplerate_pop, 'String');
    val = find(strcmp(num2str(Data.ai.SampleRate), str));
    set(handles.samplerate_pop, 'Value', val);

    % Set blocksize popup
    str = get(handles.blocksize_pop, 'String');
    val = find(strcmp(num2str(Data.ai.SamplesAcquiredFcnCount), str));
    set(handles.blocksize_pop, 'Value', val);

    % Set bit depth
    switch Data.ai.BitsPerSample
        case 16
            set(handles.bit16_radio, 'Value', true);
        case 24
            set(handles.bit24_radio, 'Value', true);
        otherwise
            error('Wrong Bit Depth!')
    end

    % Init axes
    axes_setup
end

%%
```

```

function axes_setup
    set(Handles.kernel,...
        'XLim'      , [0 1]           ,...
        'YLim'      , [-1 1]          ,...
        'XLimMode'   , 'manual'        ,...
        'YLimMode'   , 'manual');
    set(Handles.output,...
        'XLim'      , [0 1]           ,...
        'YLim'      , [-1 1]          ,...
        'XLimMode'   , 'manual'        ,...
        'YLimMode'   , 'auto');
    set(Handles.kernel_fft,...
        'XScale'     , 'log'           ,...
        'XLimMode'   , 'auto'          ,...
        'YLimMode'   , 'auto');
    % set(Handles.output_fft,...
    %     'XScale'     , 'log'           ,...
    %     'XLimMode'   , 'auto'          ,...
    %     'YLimMode'   , 'auto');
    xlabel(Handles.kernel, 'time (seg)');
    ylabel(Handles.kernel, 'amplitude (volts)');
    xlabel(Handles.kernel_fft, 'frequency (Hz)');
    ylabel(Handles.kernel_fft, 'amplitude (dBV)');
    xlabel(Handles.output, 'time (seg)');
    ylabel(Handles.output, 'amplitude (volts)');
    % xlabel(Handles.output_fft, 'frequency (Hz)');
    % ylabel(Handles.output_fft, 'amplitude (dBV)');
end
%%
function update_fft
    % init
    step = 1/Data.ai.SampleRate;

    % Kernel FFT
    Data.pow = nextpow2(Data.kernel.ln +
        Data.ai.SamplesAcquiredFcnCount - 1);
    [padded Data.kernel.toPad] = zeropad(Data.kernel.signal, 'end',
        Data.pow);
    Data.kernel.fft = fft(padded);
    Data.kernel.fftLn = length(Data.kernel.fft);
    Data.nyquist = Data.ai.SampleRate / 2 + 1;
    Data.freq = freqspace(2^Data.pow) * (Data.nyquist);

    % Define input segment variables
    Data.input.toPad = 2^Data.pow - Data.ai.SamplesAcquiredFcnCount;
    Data.input.time = 0:step:(Data.ai.SamplesAcquiredFcnCount * step)
        - step;

    % Define output segment variables
    Data.output.time =...
        0:step:((Data.kernel.fftLn + Data.input.toPad)*step) - step;
    Data.output.overlap = zeros(1, Data.input.toPad);

    % Update information
    update_info
end
%%
function reset_data
    Data.iSegment = 1;

```

```

        Data.meanProcessing = 0;
        Data.input.segment = {};
        Data.output.segment = {};
        Data.output.overlap(:) = 0;
    end

%%
function close_fig
    delete(daqfind)
    clear all
end

%% Update all info
function update_info
    update_kernel_info
    update_input_info
    update_output_info
    update_gen_info
end

%% Update kernel information in GUI
function update_kernel_info
    set(handles.kernel_ln_txt, 'String',...
        ['Length : ' num2str(Data.kernel.ln) ' (sam) : '
        num2str(Data.kernel.time(end)) ' (sec)'])
    set(handles.kernel_topad_txt, 'String',...
        ['To Pad : ' num2str(Data.kernel.toPad)])
    set(handles.kernel_fftsize_txt, 'String',...
        ['FFT Size : ' num2str(Data.kernel.fftLn)])
    % set(handles.kernel_ln_txt, 'String',...
    % ['FFT Size : ' num2str(Data.kernel.ln)])
end

%% Update output information in GUI
function update_output_info
    set(handles.output_prevln_txt, 'String',...
        ['Length : ' num2str(Data.output.time(end)) ' (sec)'])
    set(handles.output_nowln_txt, 'String',...
        ['Length : ' num2str(Data.kernel.fftLn) ' (sam) : '
        num2str(Data.output.time(end)) ' (sec)'])
    set(handles.output_sigln_txt, 'String',...
        ['Length : ' num2str(Data.kernel.fftLn) ' (sam) : '
        num2str(Data.output.time(end)) ' (sec)'])
    set(handles.output_overlap_txt, 'String',...
        ['FFT Size : ' num2str(Data.kernel.ln)])
end

%% Update input information in GUI
function update_input_info
    set(handles.input_ln_txt, 'String',...
        ['Length : ' num2str(Data.ai.SamplesAcquiredFcnCount) ...
        ' (sam) : ' num2str(Data.input.time(end)) ' (sec)'])
    set(handles.input_topad_txt, 'String',...
        ['To Pad : ' num2str(Data.input.toPad)])
    set(handles.input_fftsize_txt, 'String',...
        ['FFT Size : ' num2str(Data.kernel.fftLn)])
    % set(handles.kernel_ln_txt, 'String',...
    % ['FFT Size : ' num2str(Data.kernel.ln)])
end

```



```

%% Update general information in GUI
function update_gen_info
    set(handles.gen_processing_txt, 'String',...
        ['Processing Time : ' num2str(Data.meanProcessing*1000) '
(ms)']);
    set(handles.gen_isegment_txt, 'String',...
        ['Segment Count : ' num2str(Data.iSegment)]);
%
    set(handles.kernel_ln_txt, 'String',...
        ['FFT Size : ' num2str(Data.kernel.ln)])
end

%%

function setforpeeking
% Define timer properties
period = 1/8;
set(Data.ai,...
    'TimerPeriod', period,...
    'TimerFcn', @startpeeking)
toPeek = period * Data.ai.SampleRate / 10;
Data.input.baseValue = -60;
Data.prevdBrms = Data.input.baseValue;

% Set axes properties
set(handles.input_axes,...
    'YLim', [Data.input.baseValue 0],...
    'YLimMode', 'manual',...
    'CLim', [Data.input.baseValue 0],...
    'NextPlot', 'replacechildren');

% Plot initial dB RMS Value
handles.dBrmsBar = bar(handles.input_axes,
    Data.input.baseValue,...
    'BarWidth', 1,...
    'BaseValue', Data.input.baseValue,...
    'FaceColor', [0 1 0]);

% Start analog input to start peeking and wait for trigger
start(Data.ai);

function startpeeking(hObject, eventdata)
% Peek data
peek = peekdata(Data.ai, toPeek)';
if isempty(peek)
    return
end

% Calculate RMS Value and convert to decibels
rms = sqrt(sum(peek.^2) / length(peek));
dBrms = 2*unit2db(rms, 1/sqrt(2));
if dBrms < Data.input.baseValue
    dBrms = Data.input.baseValue;
elseif dBrms > 0
    dBrms = 0;
elseif dBrms == Data.prevdBrms
    return
end

% Plot dB RMS Value
set(handles.dBrmsBar, 'YData', dBrms)

```

```

        end
    end

    %%
    function closeadoa
        delete(daqfind);
        close(Handles.mainFig);
        clear all
    end

    %%

end

```

#### 4.1.4.5 deconvolve.m

En deconvolve.m se ejecuta el cálculo de la deconvolución de el segmento de entrada actual con el kernel ya predefinido. Sus entradas, como las de cualquier Callback, son hObject y eventdata. No tiene argumentos de salida ya que sus cálculos son acumulados en Data, la cual es una variable global.

Esta función realiza sus cálculos según los siguientes pasos:

- Inicializa el cronómetro de tiempo de procesamiento.
- Adquiere el segmento de la entrada correspondiente.
- Rellena el segmento de entrada con la cantidad de ceros necesarios para que este adquiriera el mismo tamaño de la DFT del kernel y calcula su DFT.
- Realiza la división de la DFT de la entrada sobre la DFT del kernel para calcular el espectro de la deconvolución.
- Convierte el espectro de la deconvolución al dominio temporal mediante la transformada inversa de Fourier.
- Superpone la primera parte de la serie temporal adquirida en el segmento actual con la última parte de la serie temporal adquirida en el segmento anterior (según corresponde al método Overlap-Add).

- Guarda los segmentos obtenidos para la entrada y la deconvolución.
- Escala (basado en la sumatoria de los elementos de kernel) y envía los datos válidos de la deconvolución por la salida análoga especificada.
- Finalmente, calcula el tiempo promedio de procesamiento y actualiza la información a mostrar en la GUI.

```
function deconvolve(hObject, eventdata)

global Data Handles

%% Get input segment and invert it
tic
[input Data.input.time] = getdata(Data.ai,...
    Data.ai.SamplesAcquiredFcnCount);
input = input';
Data.input.time = Data.input.time';

% Perform input segment FFT
padded = zeropad(input, 'end', Data.pow);
inputFFT = fft(padded);

% Perform deconvolution  $y = x*h \rightarrow X = Y/H$ 
outputFFT = inputFFT ./ Data.kernel.fft;
output = ifft(outputFFT);

% Overlap previous segment with current segment
output(1:Data.input.toPad) = ...
    output(1:Data.input.toPad) + Data.output.overlap;
Data.output.overlap = output(end - Data.input.toPad + 1:end);

% Save input and output segments
Data.input.segment(Data.iSegment) = {input};
Data.output.segment(Data.iSegment) =
{output(1:Data.ai.SamplesAcquiredFcnCount)};

% Sound output data
putdata(Data.ao,...
    [Data.output.segment{Data.iSegment}'... Channel #1 output
    Data.output.segment{Data.iSegment}']) % Channel #1 output
switch Data.ao.Running
    case 'Off'
        % start(Data.ao)
    end
% soundsc(Data.output.segment{Data.iSegment}, Data.ai.SampleRate,
Data.ai.BitsPerSample)

% Show and calculate mean Processing time and segment count
Data.meanProcessing = linavg(toc, Data.meanProcessing, Data.iSegment);
set(handles.gen_processing_txt, 'String',...
    ['Processing Time : ' num2str(Data.meanProcessing*1000) ' (ms)']);
set(handles.gen_isegment_txt, 'String',...
    ['Segment Count : ' num2str(Data.iSegment)]);
Data.iSegment = Data.iSegment + 1;
```

#### 4.1.4.6 zeropad.m

Esta función rellena su primer argumento de entrada, el cuál debe ser un vector, con ceros hasta que este sea del tamaño de la siguiente potencia de dos con respecto a su tamaño actual, o lo rellena hasta que alcance el tamaño de una potencia de dos especificada. Su segundo argumento de entrada (opcional) puede ser alguna de las siguientes líneas: 'split', que rellena de ceros el vector tanto al comienzo como al final; 'begin', que solo agrega ceros al comienzo del vector; 'end' que rellena el vector con los ceros al final de este. Un tercer argumento, puede ser un entero que representa la potencia de dos que determinaría el tamaño del vector de entrada. Su salida será el vector con los ceros adicionados.

```
function varargout = zeropad(varargin)
% ZEROPAD
% padded = zeropad(data)
% zeropad pads data with zeros to make data's length a power of two.
%   There are three working modes:
%       'split' = This is the default mode. It adds the half of the
zeros
%       to be padded to the beggining and then it adds the other half
at
%       the end of the data.
%       'begin' = It adds the zeros to pad at the beggining of the
data.
%       'end' = It adds the zeros to pad at the end of the data.

switch nargin
case 0
    error('Not enough input arguments!')
case 1
    data = varargin{1};
    mode = 'split';
    pow = nextpow2(length(data));
case 2
    data = varargin{1};
    mode = varargin{2};
    pow = nextpow2(length(data));
case 3
    data = varargin{1};
    mode = varargin{2};
    if varargin{3} < nextpow2(length(data))
        error('Exponent N should be greater or equal than next
power of 2');
    else
        pow = varargin{3};
```

```

        end
    otherwise
        error('Too many input arguments!')
    end
end

toPad = 2^pow - length(data);
toAdd = zeros(1, toPad);

switch mode
    case 'begin'
        padded = [toAdd data];
    case 'end'
        padded = [data toAdd];
    case 'split'
        switch mod(toPad, 2)
            case 1
                half = round(length(toAdd) / 2);
                padded = [toAdd(1, 1:half - 1) data toAdd(1, 1:half)];
            case 0
                half = length(toAdd) / 2;
                padded = [toAdd(1, 1:half) data toAdd(1, 1:half)];
            otherwise
                error('Wrong module!')
        end
    end
    otherwise
        error('Wrong padding mode!')
end

switch nargout
    case 0
        error('Not enough output arguments!')
    case 1
        varargout{1} = padded;
    case 2
        varargout{1} = padded;
        varargout{2} = toPad;
    otherwise
        error('Too many output arguments!')
end

```

#### 4.1.4.7 linavg.m

A pesar de su sencillez programática, la función `linavg.m` resulta muy útil ya que facilita la lectura del código y no toca recordar o consultar la ecuación del promediado lineal. Ya que con cada segmento de procesamiento se obtiene un nuevo valor a promediar, esta ecuación no necesita de todos los datos a la vez, sino que puede operar con cada valor nuevo y el promedio anterior. Sus argumentos de entrada son: `data`, que es el dato o el vector de datos a promediar; `prevAvg`, que es el promedio anterior; `i`, que es el

contador de cuantos promedios han sido calculados incluyendo el actual. Su salida será el promedio calculado.

```
function [avg, prevAvg] = linavg(data, prevAvg, i)

    avg = (1 / i) * data + ((i - 1) / i) * (prevAvg);

end
```

#### 4.1.4.8 unit2db.m

Esta función permite convertir cualquier unidad a una relación decibélica. Sus argumentos de entrada son variables y están planteados así: el primer argumento es el valor o vector de valores a convertir a decibels; el segundo argumento será la referencia, la cual puede ser un número real o una línea de texto para indicar una referencia preestablecida<sup>4</sup>. En caso de que solo haya un argumento de entrada, se asumirá como referencia el número uno. Su único argumento de salida será el valor en decibels correspondiente a las condiciones de entrada.

```
function varargout = unit2db(varargin)

    switch nargin
        case 0
            error('Not enough input arguments!!!')
        case 1
            data = varargin{1};
            REF = 1;
        case 2
            data = varargin{1};
            REF = varargin{2};
            if isa(REF, 'char')
                REF = references(REF);
            end
        case 3

    otherwise
        error('Too many input arguments!!!')
    end
```

---

<sup>4</sup> Estas referencias preestablecidas y sus explicaciones y usos típicos han sido extraídas de wikipedia.org.

```

% Avoid log of 0
index = data == 0;
data(index) = 1e-17;

% Convert to dB
dB = 10 * log10(data / REF);

% Return values
varargout{1} = dB;

function REF = references(REF)

switch REF
% Voltage References
case 'volts'

case 'dBV'
% Voltage relative to 1 volt, regardless of impedance.
REF = 1;
case 'dBmV'
% Voltage relative to 1 millivolt, regardless of
impedance.
% Widely used in cable television networks, where the
% nominal strength of a single TV signal at the receiver
% terminals is about 0 dBmV. Cable TV uses 75 Ohms coaxial
% cable, so 0 dBmV corresponds to -48.75 dBm or ~13 nW.
REF = 1e-3;
case 'dBu'
% Voltage relative to 0.775 volts. The 'u' comes from
unloaded.
% dBu can be used regardless of impedance, but is derived
from
% a 600 Ohms load dissipating 0 dBm (1 mW).
REF = 0.775;
% Power References
case 'dBW'
% Power measurement relative to 1 Watt
% 0 dBW = +30 dBm; -30 dBW = 0 dBm; XdBW = XdBm - 30.
REF = 1;
case 'dBmW'
% Power measurement relative to 1 miliWatt.
% XdBm = XdBW + 30.
REF = 1e-3;
% Acoustic References
case 'dBSPL'
% For sound in air and other gases, relative to
% 20 micropascals (?Pa) = 20e-6 Pa, the quietest sound a
% human can hear. This is roughly the sound of a mosquito
% flying 3 metres away. This is often abbreviated to just
% "dB", which gives some the erroneous notion that "dB" is
% an absolute unit by itself.
REF = 20e-6;
case 'dBSPL liquid'
% For sound in water and other liquids, a reference
pressure
% of 1 ?Pa is used.
case 'dBA'
case 'dBC'
case 'dBD'
case 'dBSIL'

```

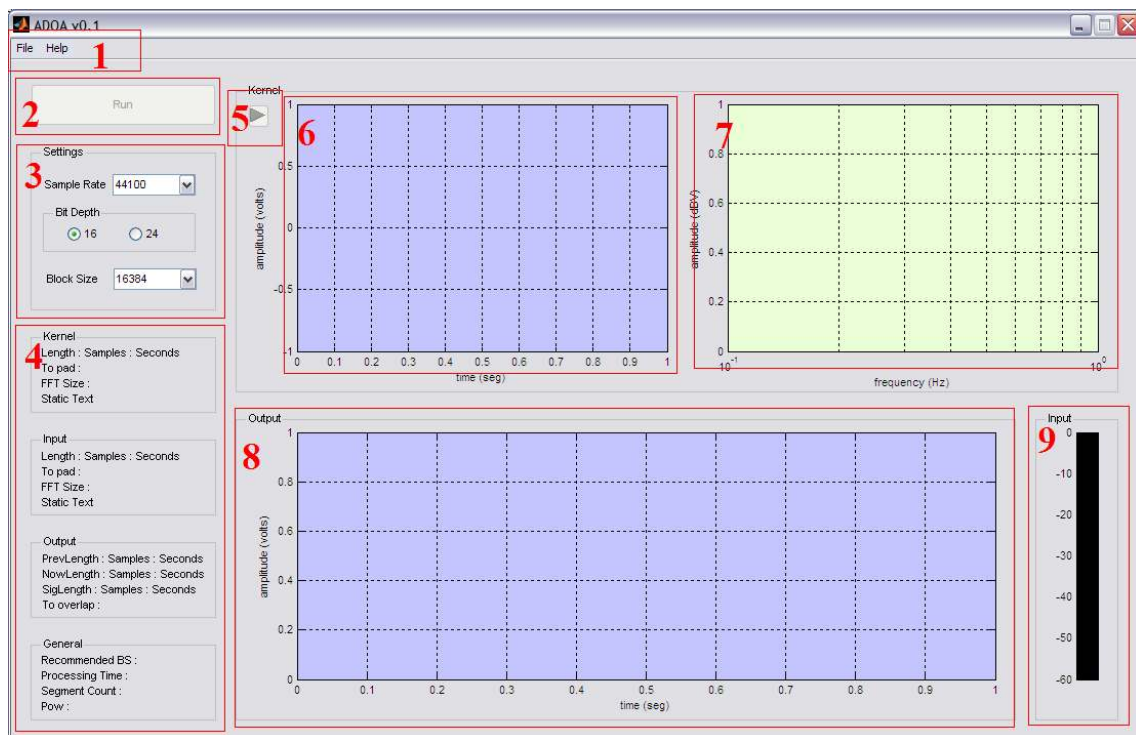
```

% Sound Intensity Level, Relative to 10e-12 W/m2, which is
% roughly the threshold of human hearing in air.
REF = 10e-12;
case 'dBSWL'
% Sound Power Level - relative to 10e-12 W.
REF = 10e-12;
end

```

#### 4.1.5 INTERFAZ GRÁFICA

Se procuró que la GUI del ADOA fuera lo más simple posible. A continuación se muestra una imagen de la pantalla principal del ADOA y se enumeran sus partes más importantes:



**Figura 26** Pantalla principal del ADOA.



1. Menús del ADOA. Los menús y sus sub-menús son los siguiente:

- File
  - Load Kernel: abre un diálogo que permite seleccionar el archivo WAV deseado que contiene el kernel a evaluar.
  - Save Unaltered: en futuras versiones, permitirá guardar en un archivo WAV la señal de entrada.
  - Save Deconvolved: en futuras versiones, permitirá guardar en un archivo WAV la señal deconvolucionada.
  - Close: permite cerrar la GUI y borra todas las variables relacionadas con el ADOA. Cumple la misma función que la X ubicada en la esquina superior derecha de la GUI.
- Help
  - Content: en futuras versiones, permitirá ver el contenido de la documentación del usuario.
  - About: en futuras versiones, permitirá ver información básica del software tal como nombre de su autor, versión, fecha de última modificación, cómo contactar al autor, etc.

2. Este botón permite correr y para el proceso de deconvolución. En esta imagen aparece como desactivado ya que ningún kernel ha sido cargado.

3. En este cuadro se pueden ajustar las siguientes propiedades de procesamiento del ADOA:

- Sample Rate es la frecuencia de muestreo de la entrada análoga.
- Bit Depth es la resolución de la entrada análoga.
- BlockSize es el tamaño del bloque de procesamiento, es decir, la cantidad de muestras que se deben adquirir por cada segmento de procesamiento.

4. Este panel permite ver información acerca del kernel cargado, la entrada análoga, la salida análoga y alguna información genérica. La mayoría de la información que se muestra en esta sección ha sido añadida a la GUI con propósitos de depuración programática y no son de gran utilidad para el usuario.
5. Este botón permite escuchar la respuesta al impulso cargada. En la imagen aparece inactivo porque ningún kernel ha sido cargado.
6. En esta sección se grafica el kernel en el dominio temporal una vez este ha sido cargado.
7. En esta sección se muestra el kernel en el dominio espectral (i.e. la DFT de su serie temporal) una vez este ha sido cargado.
8. En esta sección se muestra la deconvolución obtenida por el software una vez se ha terminado el procesamiento a petición del usuario.
9. Aquí se grafican los niveles de entrada.

## ***4.2 METODOLOGÍA DE MEDICIONES***

### **4.2.1 INTRODUCCIÓN**

Debido a las diversas posibilidades de aplicación del algoritmo de deconvolución, será necesario hacer mediciones de sistemas diferentes (i.e. acústicos, electrónicos) y de sistemas de condiciones diferentes (i.e. recintos reverberantes, recintos cuasi-anecoicos) para poder evaluar de forma concisa la precisión y eficiencia del algoritmo. Esto implica un alto rigor en la metodología de las mediciones y en la elección de los sistemas a medir, para minimizar los “errores de medición” reflejados en el algoritmo.

#### **4.2.2 LISTADO DE EQUIPOS**

Los equipos utilizados en las mediciones de sistemas eléctricos y acústicos fueron los siguientes:

- Interfaz de grabación FireWire Tascam FireOne
- Potencia Creative iTrigue 2300
- Audífonos de monitoreo Sennheiser HD280 Pro

Adicionalmente, los siguientes equipos fueron empleados sólo en las pruebas acústicas:

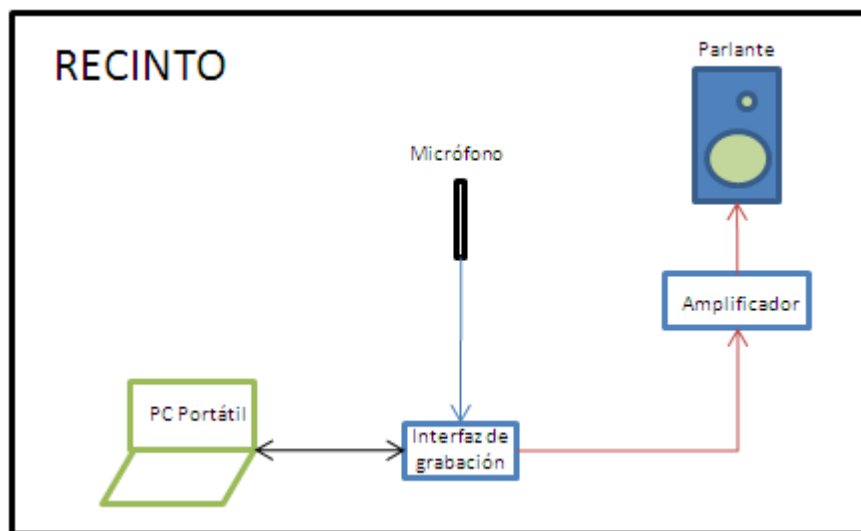
- Micrófono de mediciones omnidireccional Behringer ECM8000
- Parlante de uso genérico JBL Control1X
- Stand para micrófonos PROEL

#### **4.2.3 SISTEMAS ACÚSTICOS: RECINTOS**

Para las mediciones en recintos acústicos se planteo la siguiente metodología:

- Se graba un segmento de audio y se guarda en un archivo WAV el cual será utilizado como estímulo del recinto una vez hallada la respuesta al impulso del mismo
- Se genera un archivo WAV de ruido rosa con una duración de 30 segundos y este se utiliza como referencia para hallar la respuesta al impulso

- Se escoge una relación micrófono-fuente fija dentro del recinto y esta permanece invariable durante toda la medición
- Se reproduce el archivo con ruido rosa dentro del recinto a través del parlante y se capta la señal con el micrófono de medición, esta señal se deconvoluciona con el ruido rosa sin las alteraciones del sistema para obtener su respuesta al impulso
- Finalmente, se reproduce el estímulo con el segmento de audio en el recinto a través del parlante y se capta la señal por medio del ADOA, al cual ya se le ha cargado la respuesta al impulso hallada anteriormente, para ser deconvolucionada



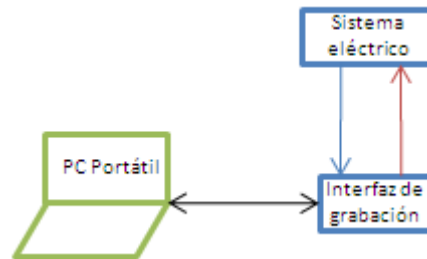
**Figura 27** Esquema de mediciones acústicas.

#### 4.2.4 SISTEMAS ELECTRÓNICOS: EQUIPOS ELECTRÓNICOS

Las mediciones de los sistemas eléctricos sigue la misma metodología de las mediciones acústicas con las siguientes excepciones:

- El parlante es descartado y la señal de prueba es inyectada directamente de la salida de la interfaz de audio a la entrada del sistema eléctrico

- El micrófono es eliminado de la cadena de medición y la señal saliente del sistema eléctrico es introducida directamente a la interfaz de audio



**Figura 28** Esquema de mediciones de sistemas eléctricos.

## 5 PRESENTACIÓN Y ANÁLISIS DE RESULTADOS

### 5.1 INTRODUCCIÓN

Debido a la naturaleza de los diferentes sistemas medidos y sobre los cuales se realizaron las pruebas de deconvolución con el ADOA, se han dividido los resultados en dos categorías, sistemas eléctricos y sistemas acústicos. Sin embargo, para minimizar las diferencias entre los posibles resultados entre ambos tipos de sistemas, se escogió un segmento de audio (*jazz line.wav*), correspondiente a una frase melódica generada por una guitarra, para estimular tanto el sistema eléctrico, como el sistema acústico.

### 5.2 SISTEMAS ELÉCTRICOS

El sistema eléctrico que se muestra a continuación corresponde a una unidad de retraso digital *DigiTech Vocal 300 Vocal Effects Processor*, con la siguiente configuración: 2 taps de retraso a 250ms y 500ms con ganancias de -3dB y -6dB respectivamente. En la Figura 29 se puede ver la respuesta al impulso calculada mediante la deconvolución del ruido rosa a la salida del sistema y el ruido rosa antes de entrar a este.

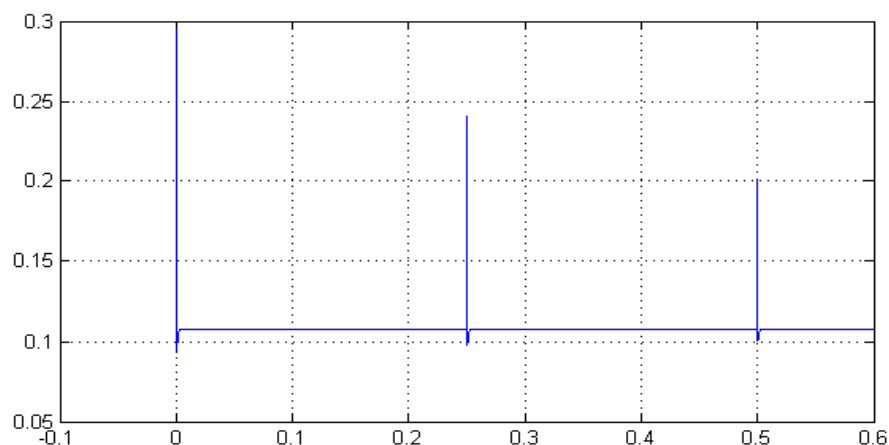
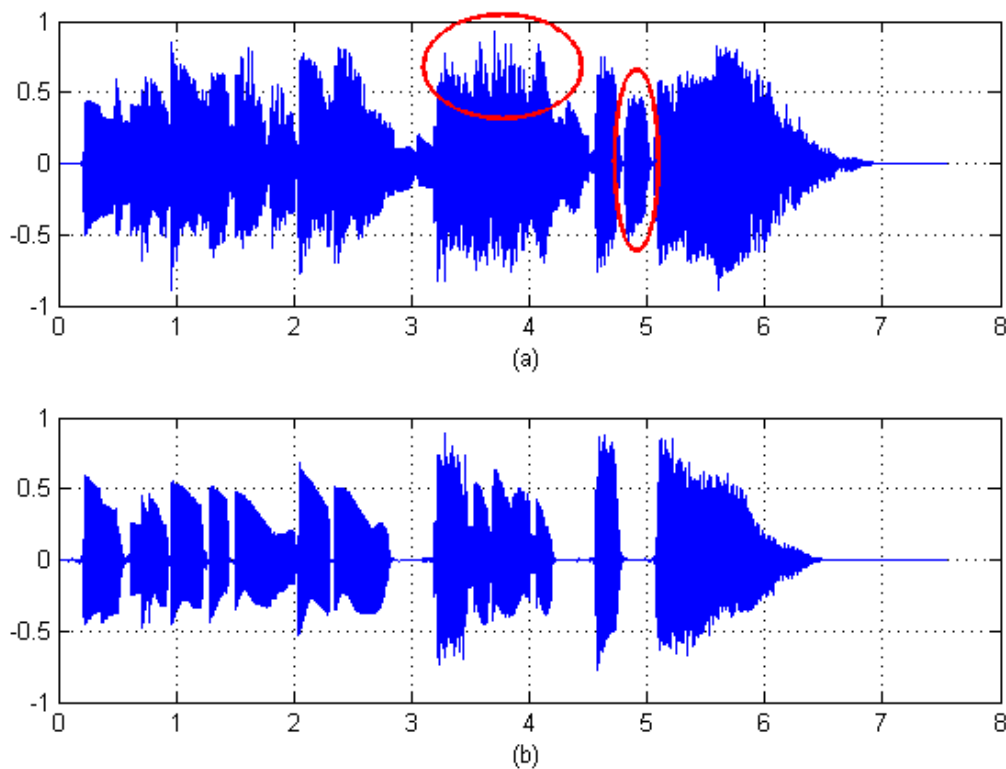


Figura 29 Respuesta al impulso del sistema de retrasos.

Luego, se excitó el procesador de retrasos con el segmento de audio *jazz line.wav* y se captó la salida del sistema con el ADOA para deconvolucionarla con la respuesta al impulso ya obtenida. La Figura 30 (a) muestra la salida de dicho sistema antes de ser procesada por el ADOA. Se puede observar en las anotaciones que se han hecho en la figura, la influencia del sistema de retraso sobre la señal original. En la Figura 30 (b) se muestra el resultado arrojado por el ADOA, es decir, la señal filtrada por la deconvolución y es notable que en comparación a la Figura 30 (a), tiene un contenido energético menor debido a la eliminación de las repeticiones.

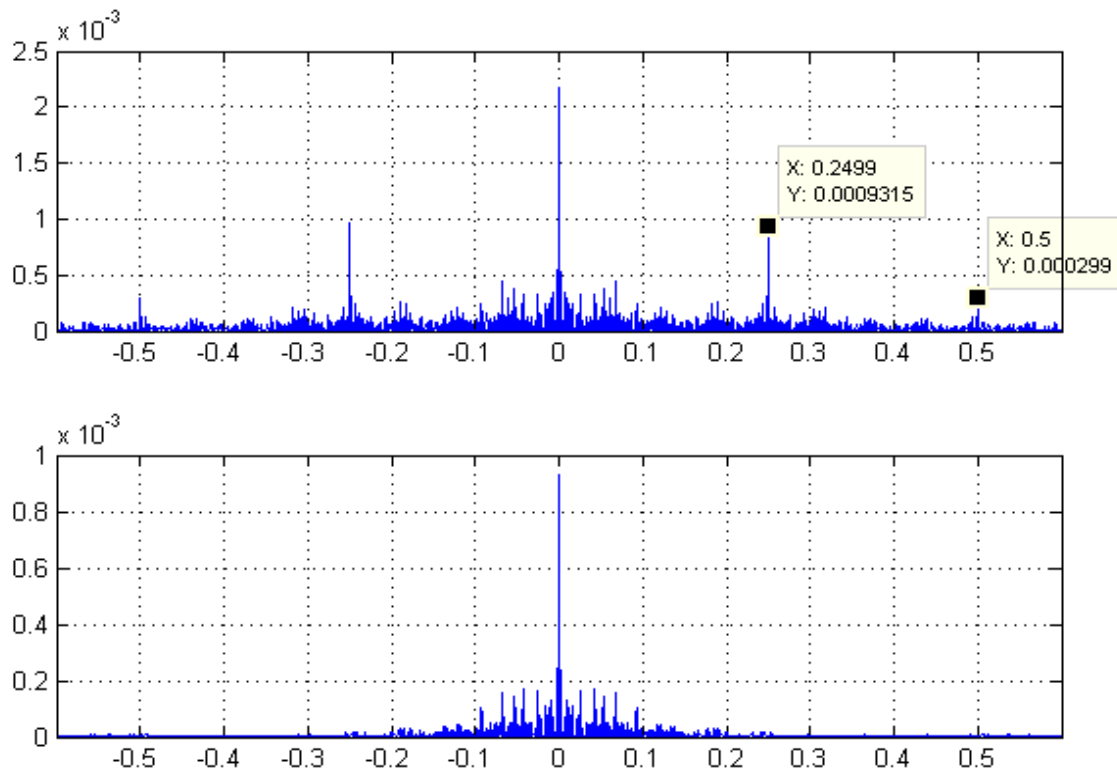


**Figura 30** Señal alterada por el sistema de retraso (a) y su versión deconvolucionada (b).

En la Figura 31 se hace la comparación de las auto-correlaciones<sup>5</sup> de la señal retrasada con la señal filtrada con el propósito de esclarecer la presencia y el filtrado de los

<sup>5</sup> La auto-correlación de una señal indica la similitud de esta con una versión desplazada en el tiempo de si misma

retrasos. Podemos observar que los retrasos han sido filtrados en la señal procesada por el ADOA de forma eficiente.



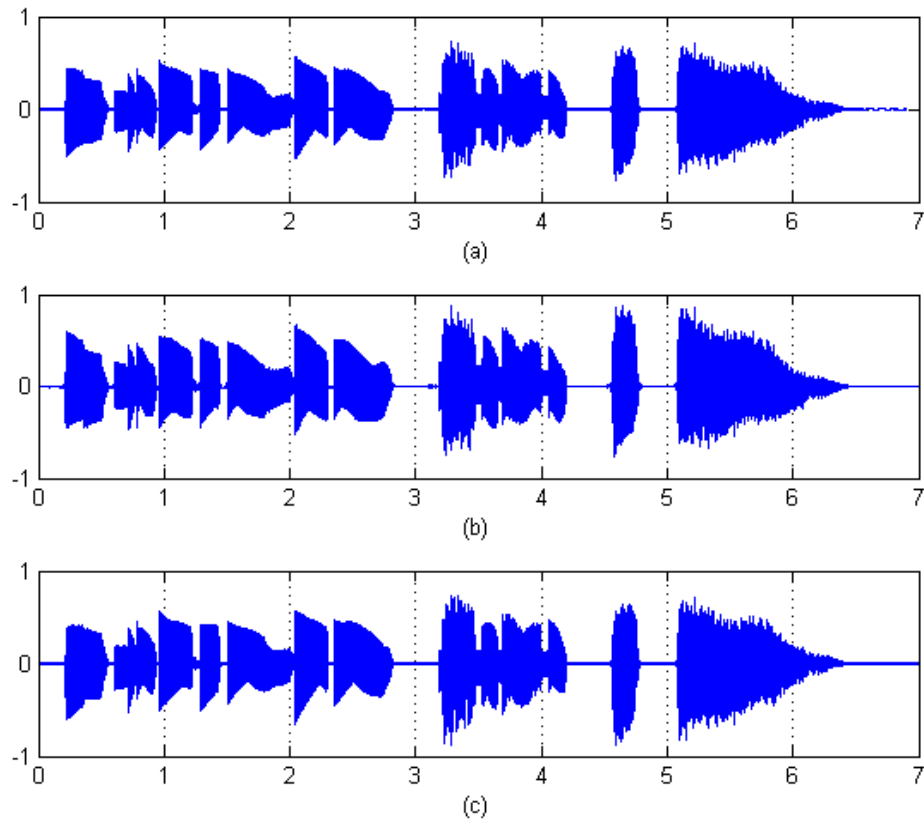
**Figura 31** Auto-correlación de la señal retrasada (arriba) y de la señal filtrada por el ADOA (abajo).

Sin embargo, en la prueba auditiva, que consistió en escuchar detalladamente las diferencias entre la señal deconvolucionada por el ADOA y la señal del archivo *jazz line.wav*, reveló que la señal procesada por el algoritmo contenía algunos ruidos irregulares (sobre todo en frecuencias altas, a partir de 4kHz) que no estaban presentes en la señal original, por lo que resultó necesario filtrar la señal del ADOA con un filtro pasa-bajo con una frecuencia central en 4kHz.

En la Figura 32 están graficadas en el dominio temporal *jazz line.wav*, la señal procesada por el ADOA y esta última filtrada con un pasa-bajo. Se puede observar que el resultado filtrado es bastante similar a la señal original, además, la prueba auditiva



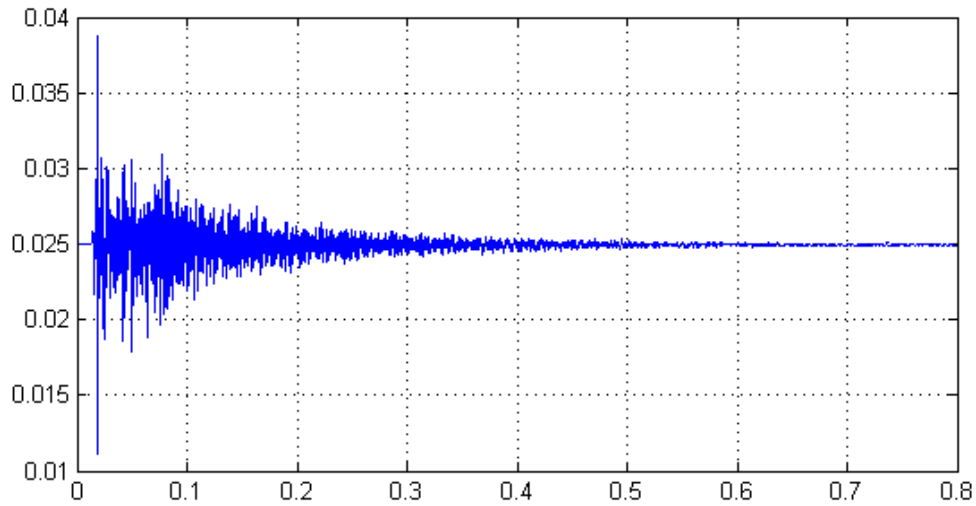
condujo a resultados muchos más satisfactorios una vez filtrada la señal deconvolucionada.



**Figura 32** Señal original (a), señal resultante del ADOA (b) y señal del ADOA filtrada con un pasa-bajo (c).

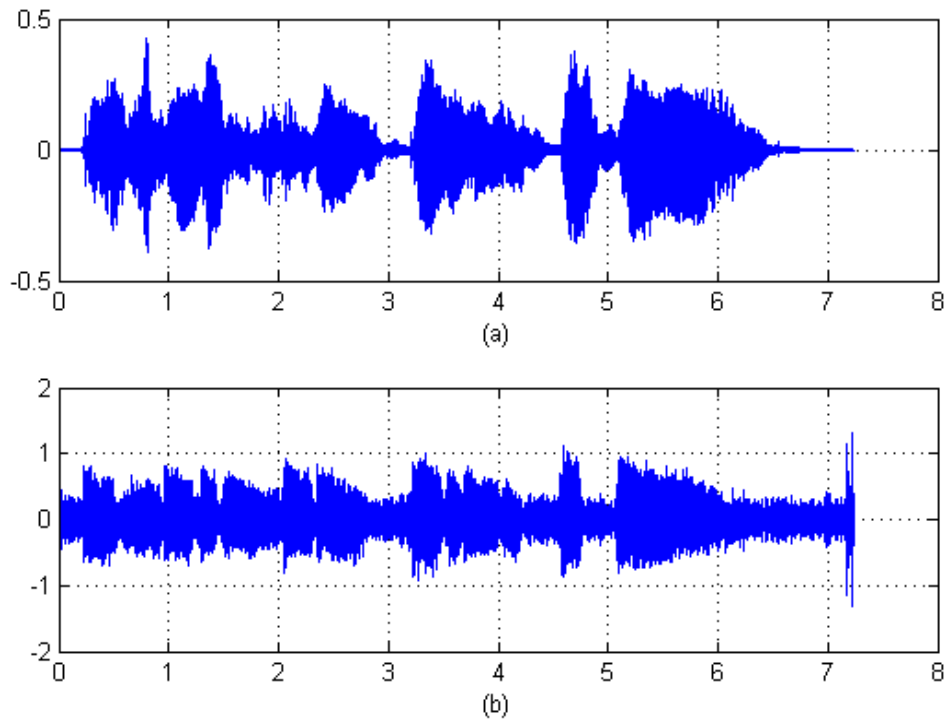
### 5.3 SISTEMAS ACÚSTICOS

Para la ilustración del funcionamiento del ADOA en recintos, se escogieron los resultados de las mediciones y pruebas realizadas en una habitación vacía de aproximadamente 7.2 x 8.5m. Se escogió este recinto ya que tiene unos tiempos de reverberación considerables (0.92 segundos en 1kHz), lo cual haría más evidente la efectividad de los resultados de la deconvolución.



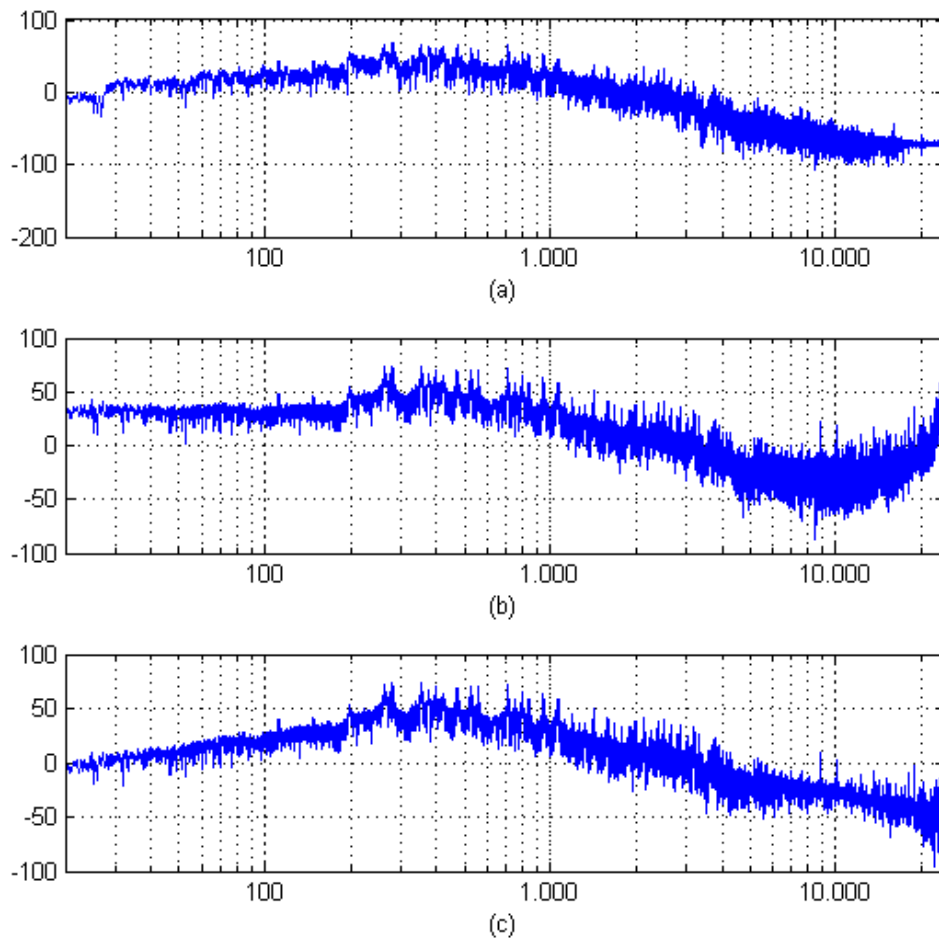
**Figura 33** Respuesta al impulso del sistema acústico evaluado.

En la Figura 33 se puede ver la respuesta al impulso de la habitación y en la Figura 34 (a) se pueden ver la señal captada dentro de este, ante el estímulo de *jazz line.wav*, con su deconvolución (b).



**Figura 34** Señal captada dentro del recinto (a) y señal procesada por el ADOA (b).

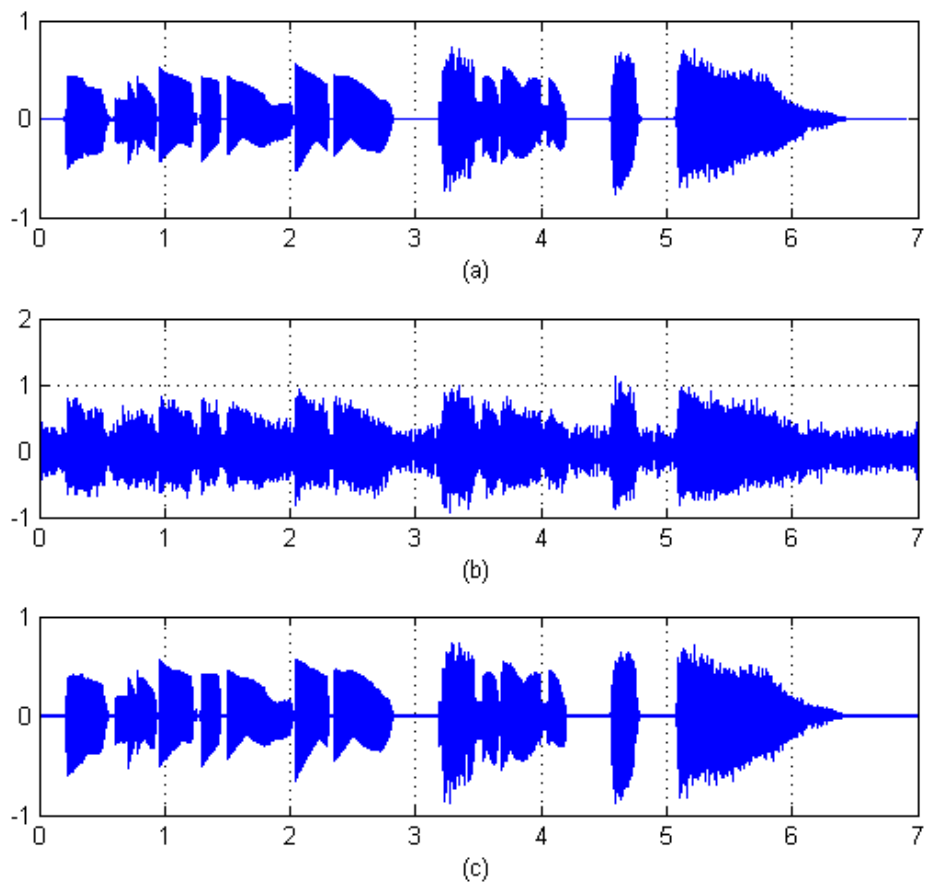
En este caso, en comparación a los resultados obtenidos en el sistema eléctrico, el resultado del ADOA es bastante confuso debido a un aumento en el ruido de fondo y a bastantes irregularidades (como la que se encuentra poco después de los 7 segundos) y a simple vista no es posible determinar la efectividad del algoritmo. La prueba auditiva fue mucho más diciente que los resultados gráficos, en la cual se puede escuchar un aumento considerable del ruido de fondo, pero además es notable que dicho ruido es principalmente en frecuencias altas, como en el caso del sistema eléctrico, solo que el aumento en este caso fue muy superior. Debido a los resultados obtenidos en la prueba subjetiva por audición, se decidió realizar un análisis espectrales de las señales involucradas en la prueba de deconvolución (ver Figura 35) para corroborar los resultados de la prueba auditiva.



**Figura 35** Espectros de la señal original de entrada (a), la señal calculada por el ADOA (b) y la señal del ADOA pasada por un filtro pasa-bajo (c).

La Figura 35 (b) muestra claramente componentes no correspondientes a la composición espectral de la señal no alterada por el sistema, Figura 35 (a). Por esta razón, se aplicó un filtro pasa-bajo con una frecuencia de corte (i.e. -3dB) en 4kHz Figura 35 (c).

En la gráfica espectral no es claramente visible el filtrado de la deconvolución, por lo que en la Figura 36 se muestran los resultados en el dominio temporal. La parte (a) de la figura corresponde a la señal sin alteraciones del sistema, la parte (b) al resultado del ADOA y la parte (c) a la señal del ADOA filtrada con el pasa-bajo mencionado anteriormente.



**Figura 36** Señales en el dominio temporal, señal original (a), señal del ADOA (b) y señal del ADOA filtrada por un filtro pasa-bajo (c).

En esta figura se puede ver con más claridad que el resultado del ADOA, al eliminársele las frecuencias altas, es muy similar a la señal sin alteraciones. La prueba auditiva demostró que eventualmente el ruido excesivo de la deconvolución había sido filtrado casi que en su totalidad y exhibió un alto grado de eficiencia o lo que es lo mismo, la reverberación del recinto fue filtrada en su totalidad aunque con aumento mínimo del ruido de fondo incluso después del filtrado de las frecuencias por encima de 4kHz.

#### **5.4 ANÁLISIS DE RESULTADOS**

Aunque la deconvolución presentó en ambas pruebas un resultado efectivo, también demostró que es capaz de inducir irregularidades que no pertenecen a ninguna de las señales involucradas en el proceso de deconvolución. Esto es debido a las características no lineales de los sistemas. En el caso del sistema eléctrico, la principal causa de ***no linealidad*** es el ruido eléctrico y en el caso del sistema acústico entran a jugar otros factores como el ruido acústico de fondo y las no linealidades de los transductores adicionales (i.e. micrófono de medición y el parlante) además del ruido eléctrico propio de toda la cadena de medición y grabación.

En general, la deconvolución probó su alta efectividad como filtro temporal, pero también se determinó que el uso de filtros y procesos adicionales es requerido para obtener resultados óptimos.

Para obtener resultados mucho más confiables, es necesario el uso de recursos con los cuales no se contaba para hacer las pruebas pertinentes. Por ejemplo, en el caso del recinto acústico, se necesitaría un cuarto altamente aislado, tanto mecánica como acústicamente, para garantizar la máxima relación señal-ruido.

## 6 CONCLUSIONES

La deconvolución FFT demuestra ser un método eficiente y efectivo para hallar la entrada de un sistema a partir de la salida y la respuesta al impulso de este. Este proceso puede ser utilizado para la recuperación de señales bajo la influencia de un sistema no deseado, como por ejemplo, un discurso en donde la inteligibilidad de la palabra es mínima debido a la reverberación y reflexiones tempranas del recinto. Si bien este es el uso que se planteó desde el comienzo de esta investigación, no es el único, ya que en base a las pruebas realizadas nos hemos dado cuenta que también es posible filtrar otro tipo de sistemas, tales como sistemas eléctricos que someten la señal de entrada a retrasos severos.

Debido a que la deconvolución opera basada en la respuesta al impulso de un sistema, que se asume o se puede aproximar a un sistema LTI, constituye un filtro flexible capaz de adaptarse a cambios y realizar filtrados que de otra forma serían imposibles o poco prácticos de implementar.

Contrario a las respuesta al impulso medidas para evaluar o diagnosticar diferentes tipos de sistemas, la deconvolución no requiere sino que las condiciones de la respuesta al impulso medida sean lo más exactas posible a las condiciones durante el procesamiento de las señales involucradas. Por ejemplo, cuando se quieren evaluar las características acústicas de un recinto a partir de su respuesta al impulso, es importante obviar o excluir todos los demás elementos de la cadena. Es decir, se quiere obtener la respuesta al impulso del recinto, sin la influencia de los parlantes o de los amplificadores o de los micrófonos. En cambio, para la deconvolución es de gran importancia considerar dentro de la respuesta al impulso, todos los elementos de la cadena (i.e. interfaces de grabación, cables, micrófonos, parlantes, etc.), y es por esto que las características requeridas de los equipos de medición tienen que ver con su linealidad e invariancia en el tiempo (i.e. alta relación señal-ruido). Por esta razón, dentro de la

cadena de medición, el micrófono puede ser un micrófono que añada coloración a la señal, siempre y cuando este sea el mismo utilizado en la deconvolución y en la medición de la respuesta al impulso.

Sin embargo, el método de deconvolución presenta varios inconvenientes. El primero y más importante, es que debido a que está basada en la conjetura de que el sistema es LTI, y como hemos mencionado anteriormente esto es muy difícil de encontrar en la naturaleza o en los sistemas físicos reales, cualquier no linealidad o variación de las propiedades del sistema en el transcurso del proceso, pueden comprometer seriamente, al punto de ser inutilizables, los resultados obtenidos, por lo que se deben tener en cuenta las irregularidades del sistema a medir y tratar de minimizarlas para aproximar el sistema a un modelo LTI.

El segundo inconveniente es que para que sea exitosa la deconvolución, es necesario medir la respuesta al impulso del sistema, lo cual no siempre resulta práctico o posible en algunos casos.

El tercer inconveniente tiene que ver con la necesidad de procesos adicionales para obtener resultados válidos a partir de la deconvolución obtenida por el ADOA, lo cual incrementa el costo computacional global del algoritmo ya que este filtrado adicional dependerá de las características y del tipo de la señal de entrada al sistema, lo que requeriría filtros adaptativos, de alto costo aritmético, para terminar de limpiar la señal deconvolucionada.

## 7 RECOMENDACIONES

Probablemente la recomendación más importante es disminuir o eliminar (de ser posible) las características no lineales del sistema a medir y garantizar la estabilidad de sus características durante la medición. Esto puede ser difícil de lograr, pero se dan las siguientes pautas para mejorar los resultados obtenidos por la deconvolución:

- La relación señal-ruido debe ser lo más alta posible. Mientras más alta esta relación, menor importancia cobrará el ruido eléctrico o acústico del sistema. Para lograr esto se recomienda lo siguiente:
  - Utilizar la mayor resolución disponible y compatible con la cadena de medición. Se sugiere utilizar un mínimo de 24 Bits de resolución.
  - Grabar las señales a los niveles más altos posibles teniendo la precaución de no saturar la señal en ningún momento ya que esto sería asumido como una no linealidad por el sistema debido a la introducción de armónicos.
  - Se debe cuidar la ganancia de los preamplificadores ya que estos pueden introducir grandes cantidades de ruido disminuyendo la relación señal-ruido.
  - Utilizar los mejores preamplificadores posibles, ya que de estos dependen en gran parte el ruido introducido a toda la cadena de medición.
  - En caso de tratarse de un sistema acústico, el recinto a medir debe tener el menor ruido de fondo acústico posible. Esto se lograría dentro de un recinto amortiguado mecánicamente y aislado acústicamente o en un recinto lejos de cualquier fuente sonora de influencia importante.



- Garantizar que las condiciones del sistema permanezcan estables durante toda la medición. Con esto se pretende acercar el sistema a la invariabilidad en el tiempo. A continuación se mencionan algunas posibilidades para conseguirlo:
  - Una vez cuadrado el sistema (i.e. ganancias, cableado, posiciones de receptor y emisor, etc.) debe permanecer exactamente igual hasta que se termine la deconvolución. Esto incluye desde la medición de la respuesta al impulso hasta la deconvolución propiamente.
  - Controlar la temperatura, humedad y otras características propias del sistema, independientemente de la naturaleza del mismo. Esto es debido a que el sistema, al cambiar sus características, cambia su respuesta al impulso y su comportamiento en general. Por ejemplo, un amplificador varía su factor de amplificación dependiendo de la temperatura de sus componentes internos.
- Minimizar corrientes DC dentro del sistema de medición ya que para que un sistema se considere como lineal su salida para una entrada de valor cero, debe ser cero.

## 8 BIBLIOGRAFÍA

- [1] *Oppenheim Alan V. y Willsky Alan S., Señales y Sistemas, Prentice Hall, 1983.*
- [2] *Harris Cyril M., Manual de Mediciones Acústicas y Control del Ruido, McGraw Hill, 1995.*
- [3] *Everest F. Alton, The Master Handbook of Acoustics, McGraw Hill, 2001.*
- [4] *Smith Steve W., The Scientist and Engineer's Guide to Digital Signal Processing, www.dspguide.com, 1997.*
- [5] *Red Salud Chepén, Mini-manual del Otorrino, <http://www.redsaludchepen.gob.pe/publicaciones/>, 2008.*
- [6] *Bellingham Matt, Room Acoustics, [http://www.postaudio.co.uk/education/acoustics/room\\_acoustics.html](http://www.postaudio.co.uk/education/acoustics/room_acoustics.html), 2006.*
- [7] *Manual de Aislamiento en la Edificación, ISOVER, 1983.*
- [8] *Loudness, <http://en.wikipedia.org/wiki/Loudness>, Wikipedia, 2008.*
- [9] *Weighting, <http://en.wikipedia.org/wiki/Weighting>, Wikipedia, 2008.*
- [10] *Sound Pressure, [http://en.wikipedia.org/wiki/Sound\\_pressure](http://en.wikipedia.org/wiki/Sound_pressure), Wikipedia, 2008.*

## 9 ANEXOS

### 9.1 *graficos.m*

Este anexo contiene el código de fuente que se utilizó para generar las gráficas que aparecen a lo largo de la investigación a excepción de las que están referenciadas y de las gráficas del análisis de resultado, ya que estas últimas fueron generadas por el ADOA (aunque algunas fueron procesadas después y graficadas por aparte).

```
% Comparación señal tiempo discreto con señal tiempo continuo
[x t] = signalgen(100, 50, 5);

subplot(2, 1, 1); plot(t, x);
xlabel('Tiempo (segs)')
ylabel('Amplitud (voltios)')

subplot(2, 1, 2); stem(x);
xlabel('Muestras')
ylabel('Amplitud (voltios)')

% Datos variación del dolar 24Abril2008 Portafolio.com.co
primerDia = {...
    '5/2/2007' 2088.2500;
    '6/1/2007' 1887.9500;
    '7/3/2007' 1958.0000;
    '8/1/2007' 1973.6500;
    '9/3/2007' 2166.0000;
    '10/1/2007' 2008.8000;
    '11/1/2007' 2007.0000;
    '12/3/2007' 2049.0000;
    '1/2/2008' 2017.0000;
    '2/1/2008' 1923.9400;
    '3/3/2008' 1838.2500;
    '4/1/2008' 1829.0500};
ultimoDia = {...
    '5/31/2007' 1894.1000;
    '6/29/2007' 1974.0000;
    '7/31/2007' 1956.1500;
    '8/31/2007' 2166.0000;
    '9/28/2007' 2020.3000;
    '10/31/2007' 1984.0000;
    '11/30/2007' 2054.5000;
    '12/28/2007' 2019.8000;
    '1/31/2008' 1938.0000;
    '2/29/2008' 1839.1000;
    '3/31/2008' 1831.0000;
    '4/23/2008' 1762.5500};
% axes(1) = subplot(2, 1, 1);
```

```

ax = axes('YLim', [1700 2200]);
barSer = bar([primerDia{:,2}]);
ylabel('Valor TRM del dolar (COL$)')

% axes(2) = subplot(2, 1, 2);
% barSer(2) = bar([ultimoDia{:,2}]);
% title('Fluctuación del dolar (Último día del mes)')
% ylabel('Valor TRM del dolar (COL$)')

set(barSer,...
    'BaseValue', 1700,...
    'FaceColor', [0.8 0.8 0.8]);
set(ax,...
    'XTickLabel', {'May';
                   'Jun';
                   'Jul';
                   'Ago';
                   'Sep';
                   'Oct';
                   'Nov';
                   'Dic';
                   'Ene';
                   'Feb';
                   'Mar';
                   'Abr'}));
%% Sistema retroalimentado
beta = 2;
x = signalgen(100, 10, 5);
% x = ones(1, 1000);
y = zeros(1, length(x));
y(1) = beta*x(1);

for i = 2:length(x)
    y(i) = beta*(x(i) + y(i - 1));
end
plot(y)

%% invariancia en el tiempo
k = 20;
f = 60;
beta = 2;
fs = 1000;
N = 50;

n = 0:1/fs:(N - 1)/fs;

x = sin(2*pi*f*n);
xre = sin(2*pi*f*(n - (k - 1)/fs));

y = beta*x;
yre = beta*xre;

h(1) = subplot(2,1,1);plot(y); xlim([1 length(y)]);
h(2) = subplot(2,1,2);plot(yre); xlim([1 length(yre)]);

linkaxes(h, 'x');

%% variancia en el tiempo

```

```

k = 4;
x = signalgen(100, 100, 5);
xre = [zeros(1, k) x];
beta = (1:length(x)).^2;

y = beta.*x;
yre = [zeros(1, k) beta].*xre;

h(1) = subplot(2,1,1);plot(y); xlim([1 length(y)]);
h(2) = subplot(2,1,2);plot(yre); xlim([1 length(yre)]);

linkaxes(h, 'x');

%% Linealidad

%% DFT
fs = 1000;
f = [10 100 200];
N = 1000;
nyquist = N/2 + 1;
freq = freqspace(N) * (fs/2 + 1);

[x n] = signalgen(fs, N, f, [1 0.75 0.5]);

X = fft(x)*2/N;

h(1) = subplot(3,1,1);plot(n, x); xlim([n(1) n(end)]);
h(2) = subplot(3,1,2);semilogx(freq, unit2db(abs(X(1:nyquist))));
xlim([1 nyquist]);
h(3) = subplot(3,1,3);plot(freq, unwrap(angle(X(1:nyquist)))); xlim([1
nyquist]);

%% N DFT
fs = 1000;
f = [10 100 200];
N = 1000;

[x n] = signalgen(fs, N, f, [1 0.75 0.5]);

N = 1000;
X = fft(x)*2/N;
nyquist = N/2 + 1;
freq = freqspace(N) * (fs/2 + 1);

h(1) = subplot(3,1,1);semilogx(freq, unit2db(abs(X(1:nyquist))));
xlim([freq(1) freq(end)]);

N = N/2;
X = fft(x, N)*2/N;
nyquist = N/2 + 1;
freq = freqspace(N) * (fs/2 + 1);

h(2) = subplot(3,1,2);semilogx(freq, unit2db(abs(X(1:nyquist))));
xlim([freq(1) freq(end)]);

N = N/2;
X = fft(x, N)*2/N;

```

```

nyquist = N/2 + 1;
freq = freqspace(N) * (fs/2 + 1);

h(3) = subplot(3,1,3);semilogx(freq, unit2db(abs(X(1:nyquist))));
xlim([freq(1) freq(end)]);
% linkaxes(h, 'x');

%% FFT
N = 1:2^16;

plot(N, N, N, N.*log(N))

%% FFT deconv
clear
clc

fs = 512;
lh = 20;

h = zeros(1, lh);
h(5) = 1;
h(10) = 0.75;
h(15) = 0.5;

y = [ones(1, 10) zeros(1, 10)];
y = conv(y, h);

ly = length(y);
lx = ly - lh + 1;
block = ly + lh - 1;

Y = fft([y zeros(1, block - ly)]);
H = fft([h zeros(1, block - lh)]);
X = Y./H;

x = ifft(X);

figure(1)
ax(1) = subplot(3,1,1); stem(y)
ax(2) = subplot(3,1,2); stem(h)
ax(3) = subplot(3,1,3); stem(x)

linkaxes(ax, 'x')

%% conv
clear
clc

fs = 512;
lh = 20;

h = zeros(1, lh);
h(5) = 1;
h(10) = 0.75;
h(15) = 0.5;

x = [ones(1, 10) zeros(1, 10)];
y = conv(x, h);

```

```

figure(1)
ax(1) = subplot(3,1,1); stem(h)
ax(2) = subplot(3,1,2); stem(y)
ax(3) = subplot(3,1,3); stem(x)

%% fft conv
clear
clc

fs = 256;
lx = 256;
lh = 257;
N = lx + lh - 1;

x = signalgen(fs, lx, 20);
h = signalgen(fs, lh, 20);

X = fft([x zeros(1, N - lx)])/N;
H = fft([h zeros(1, N - lh)])/N;
Y = X.*H;
y = ifft(Y)*N;

figure(1)
ax(1) = subplot(3,1,1); plot(x)
ax(2) = subplot(3,1,2); plot(h)
ax(3) = subplot(3,1,3); plot(y)

freq = (-lx:fs/lx:lx - 1)/2;

figure(2)
subplot(3,2,1); plot(freq, abs(fftshift(X)))
subplot(3,2,3); plot(freq, abs(fftshift(H)))
subplot(3,2,5); plot(freq, abs(fftshift(Y)))
subplot(3,2,2); plot(freq, unwrap(angle(fftshift(X))))
subplot(3,2,4); plot(freq, unwrap(angle(fftshift(H))))
subplot(3,2,6); plot(freq, unwrap(angle(fftshift(Y))))

%% time conv
clear
clc

fs = 256;
lx = 256;
lh = 256;
block = lx + lh - 1;

x = signalgen(fs, lx, 5);
X = fft(x)/lx;
h = signalgen(fs, lh, 5);
H = fft(h)/lx;

y = x.*h;
Y = fft(y)/lx;

figure(1)
ax(1) = subplot(3,1,1); plot(x)
ax(2) = subplot(3,1,2); plot(h)
ax(3) = subplot(3,1,3); plot(y)

```

```

nyq = lx/2 + 1;
freq = -nyq + 1:fs/lx:nyq - 2;

figure(2)
subplot(3,2,1); plot(freq, abs(fftshift(X)))
subplot(3,2,3); plot(freq, abs(fftshift(H)))
subplot(3,2,5); plot(freq, abs(fftshift(Y)))
subplot(3,2,2); plot(freq, unwrap(angle(fftshift(X))))
subplot(3,2,4); plot(freq, unwrap(angle(fftshift(H))))
subplot(3,2,6); plot(freq, unwrap(angle(fftshift(Y))))

```