Universidad de Valladolid
Departamento de Informática

Proyecto de Investigación

La transformada wavelet y el análisis de señales temporales

Diego R. Llanos Ferraris
diego@infor.uva.es, http://www.infor.uva.es/ diego
Tutor: Dr. Valentín Cardeñoso Payo
8 de julio de 1998

Índice General

I La transformada wavelet **11**

1 Introducción a la transformada wavelet **13**
1.1 La Transformada Wavelet Continua 13
1.2 Análisis de señales no estacionarias 14
1.3 La transformada de Fourier de tiempo corto (STFT) 15
1.4 La Transformada Wavelet continua 18
1.5 Las nociones de escala y resolución 20

2 Cálculo y aplicaciones de la transformada Wavelet **23**
2.1 El algoritmo piramidal rápido 25
2.2 Paquetes wavelet y paquetes coseno 26
2.3 Codificación subbanda 28
2.4 Algunas aplicaciones 28
2.4.1 Compresión de datos 29
2.4.2 Aceleración de cálculos 29
2.4.3 Eliminación del ruido 29
2.4.4 Prototipado rápido de señales sísmicas 29
2.4.5 Astronomía 32
2.4.6 Estudio de flujo turbulento 32
2.4.7 Imagen 32

3 Análisis Wavelet y procesamiento de señal **33**
3.1 Introducción 33
3.2 Análisis y síntesis wavelet 33
3.3 Escalogramas 34
3.4 Tramas wavelet y bases ortonormales 36
3.4.1 Discretización de parámetros tiempo-escala 36
3.4.2 Tramas wavelet 37
3.4.3 Bases de wavelets ortogonales 38
3.5 El caso de tiempo discreto 39
3.5.1 Pirámide multirresolución 39

3.5.2 Esquemas de codificación subbanda 41
3.6 La Transformada Wavelet Discreta 44
3.6.1 Filtros iterados y regularidad 44
3.6.2 Funciones de escalado y wavelets obtenidos a partir de filtros iterados 45
3.6.3 Filtros escalados regulares 48

II Aplicaciones de la Transformada Wavelet en el análisis de señales temporales 51

4 Filtrado de señales de voz con "Wavelet Shrinkage" 53
4.1 Introducción 53
4.2 Implementación del algoritmo propuesto 55
4.2.1 Adecuación de un conjunto de muestras de voz 56
4.2.2 Desarrollo del módulo de filtrado wavelet 56
4.2.3 Evaluación de la calidad del filtrado 57
4.3 Elección del umbral de *thresholding* 58
4.3.1 Elección del umbral a partir de la energía 58
4.3.2 Elección del umbral adaptado a las componentes de la escala 59
4.3.3 Utilización del criterio *soft-thresholding* 62
4.3.4 Otros criterios de elección de umbral de filtrado 62
4.4 Elección del filtro QMF a utilizar 62
4.5 Conclusiones 64

A Implementación de funciones Matlab para filtrado de señal de voz 65
A.1 Introducción 65
A.1.1 Obtención de un conjunto de muestras de voz 65
A.1.2 Desarrollo de funciones de tratamiento de escalas wavelet 65
A.1.3 Evaluación del filtrado realizado 67
A.1.4 Elección del umbral de thresholding 68
A.1.5 Elección del filtro QMF a utilizar 68
A.2 Funciones Matlab desarrolladas 69
A.2.1 Función iso9000() 69
A.2.2 Función erroral() 69
A.2.3 Función evalvisu() 70
A.2.4 Función extrband() 70
A.2.5 Función filtband() 71
A.2.6 Función l162mat() 72

A.2.7 Función mat2l16() 73
A.2.8 Función mismoal() 73
A.2.9 Función porcene() 74
A.2.10 Función porcenex() 75
A.2.11 Función vfiltro() 76
A.2.12 Función vfiltror() 77
A.2.13 Función vfil000() 79
A.2.14 Función qmf2filtro() 80
A.2.15 Función vfilsoft() 81
A.2.16 Función vfilrout() 82
A.2.17 Función comphard() 84
A.2.18 Función espectro() 85

Índice de Figuras

1.1 Funciones básicas y resolución tiempo-frecuencia de la STFT y la WT 17
1.2 División del dominio de la frecuencia para la STFT y la WT . 19

2.1 Descomposición wavelet de una señal diente de sierra 24
2.2 Paquetes wavelet y paquetes coseno 27
2.3 Eliminación del ruido en una señal NMR 30
2.4 Reconstrucción de una señal sísmica 31

3.1 Regiones de influencia de un pulso de Dirac 35
3.2 Rejilla de muestreado diádica en el plano tiempo-escala 36
3.3 Cambios en la resolución y en la escala en tiempo discreto . . 39
3.4 Esquema piramidal 41
3.5 Esquema de codificación subbanda 42
3.6 Funciones de escalado 46
3.7 Dos escalas y desplazamientos del wavelet D4 47
3.8 Wavelet ortonormal generado a partir de un filtro regular de tamaño 18 47
3.9 Wavelets biortogonales 48
3.10 Ejemplos de filtro paso bajo iterado 49

4.1 Diagrama de bloques del método de filtrado en el dominio wavelet. 55
4.2 Detalle del módulo de filtrado. 57
4.3 Fragmento de la señal correspondiente a "*six*". 60
4.4 Señal de la figura 4.3 una vez filtrada. 60
4.5 Espectrograma de la palabra "*six*" antes del filtrado. 61
4.6 Espectrograma de la palabra "*six*" después del filtrado. 61

Índice de Tablas

4.1 Comparativa de resultados de filtrado para los diez mejores filtros de entre los 25 analizados 63

Prólogo

El presente documento se enmarca en el proyecto de investigación denominado "Aplicación de la Transformada Wavelet en el análisis de señales temporales". En dicho proyecto se ha intentado lograr una perspectiva de la utilidad de la transformada wavelet como alternativa a otros métodos tradicionales de procesamiento de señal, como la transformada de Fourier.

El trabajo consta de dos partes. En la primera, que lleva por título "La transformada wavelet", se describen brevemente los fundamentos de dicha transformada, cómo se calcula y algunas aplicaciones en las que ha demostrado su utilidad. En la segunda parte del trabajo, titulada "Aplicación de la transformada wavelet en el análisis de señales temporales", se describen las experiencias realizadas por el autor en el filtrado de señales de voz utilizando la técnica conocida como "wavelet thresholding".

Se presenta este trabajo como Memoria del Proyecto de Investigación que forma parte del programa de Doctorado "Tecnología de la Información". Ha sido realizado por D. Diego R. Llanos Ferraris, bajo la dirección del Dr. D. Valentín Cardeñoso Payo, siendo Tutor del autor de este documento el Dr. D. José Manuel Marqués Corral.

Parte I

La transformada wavelet

Capítulo 1

Introducción a la transformada wavelet

En este capítulo intentaremos hacer un bosquejo acerca de la transformada wavelet[1] y de su utilidad en el tratamiento de señal. Al tratarse de un capítulo introductorio, no se hará un análisis matemático profundo, sino que se explicará en que consiste la transformada wavelet, cuáles son los problemas que resuelve, y en qué aplicaciones se está experimentando con su uso.

1.1 La Transformada Wavelet Continua

La teoría de wavelets suministra un entorno de trabajo unificado para un conjunto de técnicas que han sido desarrolladas de forma independiente desde diferentes campos de aplicación del procesamiento de señal. De hecho, la teoría de wavelets cubre un amplio espectro. Trata tanto los casos de tiempo discreto como continuo, y suministra técnicas muy generales que pueden aplicarse a muchas tareas en procesamiento de señal, además de tener otras potenciales aplicaciones.

En particular, la Transformada Wavelet (WT) es interesante para el análisis de señales no estacionarias porque constituye una alternativa a la Transformada de Fourier Short-Time (STFT) o a la Transformada Gabor. La diferencia básica es la siguiente: En contraste con la STFT, que utiliza una única ventana de análisis, la WT utiliza ventanas cortas en altas frecuencias y ventanas largas en bajas frecuencias. Ese es la filosofía del llamado análisis "Q constante", o análisis de frecuencia constante relativo al ancho de banda.

[1] La palabra *wavelet* proviene del término francés *ondelette*, que puede traducirse como "pequeña onda". Mantendremos aquí el término anglosajón, por ser de uso muy difundido.

Para determinadas aplicaciones es deseable ver la WT como una descomposición de señales en un conjunto de funciones básicas. De hecho, las funciones básicas denominadas *wavelets* están siempre detrás del análisis wavelet. Se obtienen a partir de un único prototipo wavelet a través de dilataciones y contracciones (escalado), a la par que desplazamientos. El wavelet prototipo puede considerarse así como un filtro paso banda, y la propiedad "Q constante" de los otros filtros paso banda (wavelets) son consecuencia de ser versiones escaladas del prototipo.

Por lo tanto, en la WT, la noción de *escala* se propone como una alternativa a la frecuencia, llegando así a la *representacioón en tiempo-escala.* Esto significa que una señal se mapea dentro de un plano tiempo-escala, equivalente al plano tiempo-frecuencia utilizado en la STFT.

Existen diferentes tipos de transformadas wavelet, y, dependiendo de la aplicación, se eligen unas u otras. Para una señal de entrada continua, los parámetros de tiempo y de escala pueden ser continuos, lo que nos lleva a la Transformada Wavelet Continua (CWT). También pueden ser discretos, llegando a la expansión en Series de Wavelet. Finalmente, la transformada wavelet puede definirse para señales de tiempo discretas, llegando así a la Transformada Wavelet Discreta (DWT). En este último caso se utilizan técnicas de procesamiento de señal de tasa múltiple, y está relacionado con los esquemas de codificación subbanda utilizados en procesamiento de habla y en compresión de imágenes. Es de resaltar la analogía con la Transformada Contínua de Fourier, las Series de Fourier y la Transformada Discreta de Fourier.

1.2 Análisis de señales no estacionarias

El objeto del análisis de señales es la extracción de información relevante de una señal a través de su transformación. Algunos métodos hacen a priori suposiciones sobre la señal a analizar; esto permite obtener resultados satisfactorios si las suposiciones son válidas, pero obviamente no son de aplicación general. Consideraremos en adelante métodos aplicables a cualquier señal. De esta forma, el análisis destinado a representar la señal y a realizar operaciones como estimación de parámetros, codificación y reconocimiento de patrones puede realizarse "del lado de la transformada", en donde las propiedades de la señal pueden ser más evidentes.

Las transformadas se han estado aplicando a señales estacionarias, esto es, señales cuyas propiedades no evolucionan en el tiempo. Para esas señales $x(t)$, la *transformada estacionaria* natural es la conocida Transformada de Fourier:

$$X(f) = \int_{-\infty}^{+\infty} x(t)e^{-2j\pi ft}dt \tag{1.1}$$

Los coeficientes de análisis $X(f)$ definen la noción de frecuencia global f en una señal. Como se ve en (1.1), se calculan como el producto interior de la señal con funciones básicas senoidales de duración infinita. En consecuencia, el análisis de Fourier trabaja bien si $x(t)$ está compuesta por unas pocas componentes estacionarias (por ejemplo, senoidales). Sin embargo, cualquier cambio abrupto en el tiempo en una señal $x(t)$ no estacionaria se extiende a lo largo de todo el eje de frecuencias en $X(f)$. En consecuencia, un análisis adaptado a señales no estacionarias requiere algo más que la Transformada de Fourier.

El enfoque habitual consiste en introducir dependencia del tiempo en el análisis de Fourier mientras se preserva la linealidad. La idea es introducir un parámetro de "frecuencia local" (local en el tiempo) de forma que la Transformada de Fourier "local" se aplique a la señal a través de una ventana en la cual la señal es aproximadamente estacionaria. Otra forma equivalente es modificar las funciones senoidales básicas utilizadas en la Transformada de Fourier a funciones básicas más concentradas en el tiempo, pero menos concentrada en frecuencia.

1.3 La transformada de Fourier de tiempo corto (STFT): análisis de resolución fija

La Transformada de Fourier fue adaptada en primer lugar por Gabor para definir una representación en tiempo-frecuencia $S(t, f)$ compuesta por características espectrales dependientes del tiempo. Consideremos una señal $x(t)$, y supongamos que es estacionaria cuando la vemos a través de una ventana $g(t)$ de tamaño limitado, centrada en la localización temporal τ. La Transformada de Fourier de las señales enmarcadas[2] $x(t)g^*(t-\tau)$ produce la Transformada de Fourier de tiempo corto (STFT)

$$STFT(\tau, f) = \int x(t)g^*(t-\tau)e^{-2j\pi ft}dt \tag{1.2}$$

que "mapea" la señal en una función bidimensional en un plano tiempo-frecuencia (τ, f).

[2]Se utilizará la expresión *señales enmarcadas* como traducción del término ingés *windowed signals.*

El parámetro f en (1.2) es similar a la frecuencia de Fourier y muchas propiedades de la Transformada de Fourier se aplican a la STFT. Sin embargo, el análisis aquí depende críticamente de la elección de la ventana $g(t)$.

Una visión alternativa se basa en una interpretación de banco de filtros del mismo proceso. A una frecuencia dada f, (1.2) filtra la señal constantemente, con un filtro paso banda que tiene como impulso respuesta la función de ventana modulada a esa frecuencia. De esta forma, la STFT puede verse como un banco de filtros modulados.

A partir de esta doble interpretación puede verse una posible desventaja relativa al análisis tiempo-frecuencia. Consideremos la capacidad de la STFT para discriminar entre dos sinusoides puras. Dadas una función de ventana $g(t)$ y su transformada de Fourier $G(f)$, definimos el "ancho de banda" Δf del filtro como

$$\Delta f^2 = \frac{\int f^2 \mid G(f) \mid^2 df}{\int \mid G(f) \mid^2 df} \tag{1.3}$$

donde el denominador es la energía de $g(t)$. Dos sinusoides pueden discriminarse sólo si están separadas más de Δf (esta es una medida RMS, aunque son posibles otras medidas). Por lo tanto, la resolución en frecuencia del análisis STFT está dada por Δf. De forma similar, la amplitud en el tiempo viene dada por Δt como

$$\Delta t^2 = \frac{\int t^2 \mid g(t) \mid^2 dt}{\int \mid g(t) \mid^2 dt} \tag{1.4}$$

donde el denominador es otra vez la energía de $g(t)$. Dos pulsos en el tiempo pueden discriminarse sólo si están separados más de Δt.

Ahora bien, la resolución en tiempo y frecuencia no puede ser arbitrariamente pequeña, porque su producto está acotado inferiormente: el producto ancho de banda por tiempo cumple la siguiente relación:

$$\Delta t \Delta f \geq \frac{1}{4\pi} \tag{1.5}$$

Esta inecuación se conoce como el *principio de incertidumbre*, o la inecuación de Heisenberg. Significa que uno sólo puede obtener resolución en el tiempo a cambio de perder resolución en frecuencia, o vice versa.

Más importante aún es que, una vez que se elije una ventana para la STFT, la resolución en tiempo-frecuencia dadas por (1.3) y (1.4) se fija en todo el plano tiempo-frecuencia, ya que la misma ventana se utiliza para todas las frecuencias.

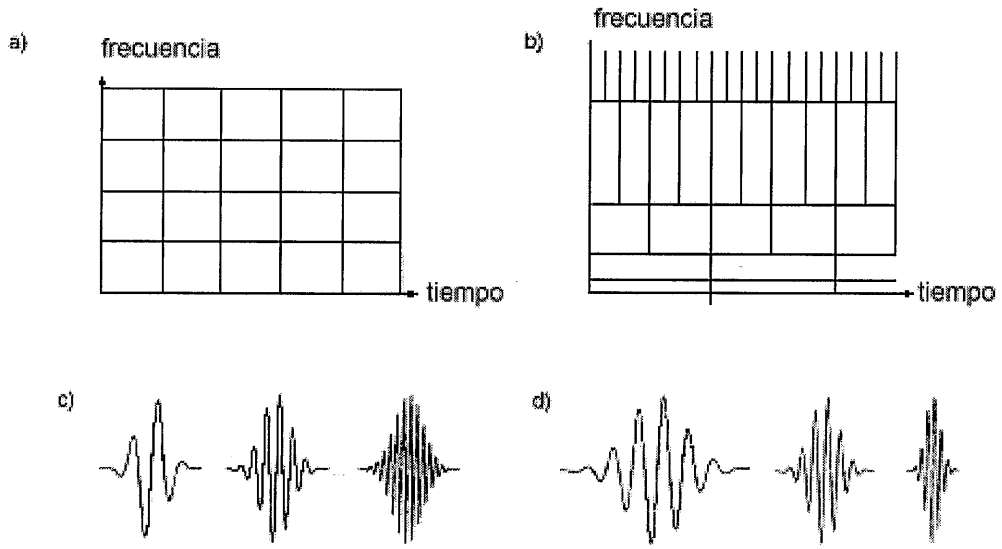

Figura 1.1: Funciones básicas y resolución tiempo-frecuencia de la STFT y la WT. Los rectángulos representan la concentración en el plano tiempo-frecuencia para cada una de las funciones bases indicadas: a) y c) para la STFT; b) y d) para la WT.

En la figura 1.1, los cuadrados representan la concentración esencial en el plano tiempo-frecuencia de una función básica dada. La figura 1.1a representa el recubrimiento del plano tiempo-frecuencia para la STFT; 1.1b representa lo mismo para la WT. 1.1c muestra las funciones básicas correspondientes para la STFT, y 1.1d muestra las funciones básicas para la WT. Por ejemplo, si la señal se compone de pequeñas ráfagas asociadas con componentes casi estacionarias, entonces cada tipo de componente puede analizarse con buena resolución en el tiempo o en la frecuencia, pero no en ambas.

1.4 La Transformada Wavelet continua: Análisis multiresolución

Para superar la limitación de resolución de la STFT, podemos imaginar que permitimos que la resolución Δf y Δt varíen en el plano tiempo-frecuencia de forma de obtener un análisis multiresolución. Intuitivamente, cuando se ve el análisis como un banco de filtros, la resolución en el tiempo debe incrementarse con la frecuencia central de los filtros de análisis. Por lo tanto, impondremos que Δf sea proporcional a f, o

$$\frac{\Delta f}{f} = c \tag{1.6}$$

donde c es una constante. El banco de filtros de análisis se compondrá entonces de filtros paso-banda con un ancho de banda relativo a una constante (también denominado "análisis Q constante"). Otra forma de decir esto es que, en lugar de que la respuesta en frecuencia del filtro de análisis se distribuya regularmente sobre el eje de frecuencia, como sucede para la STFT, se distribuye regularmente en una escala logarítmica (figura 1.2).

Esta clase de banco de filtros se utiliza, por ejemplo, para modelar la respuesta en frecuencia de la cóclea[3] situada en el oído interno. Por lo tanto, están adaptados para la percepción auditiva: por ejemplo, los filtros que satisfacen la ecuación (1.6) están distribuidos de forma natural en octavas.

Podemos ver que, cuando se satisface (1.6), Δf y por lo tanto también Δt cambian con la frecuencia central del filtro de análisis. Por supuesto, aún se satisface la inecuación de Heisemberg (1.5), pero ahora la resolución en el tiempo es arbitrariamente buena a altas frecuencias, mientras la resolución en frecuencia es arbitrariamente buena a bajas frecuencias. Por ejemplo, dos ráfagas cortas muy cercanas siempre pueden eventualmente ser separadas en

[3]Tubo espiral que forma parte del oido interno, en donde se encuentra el Órgano de Corti, encargado de convertir las vibraciones sonoras en impulsos nerviosos.

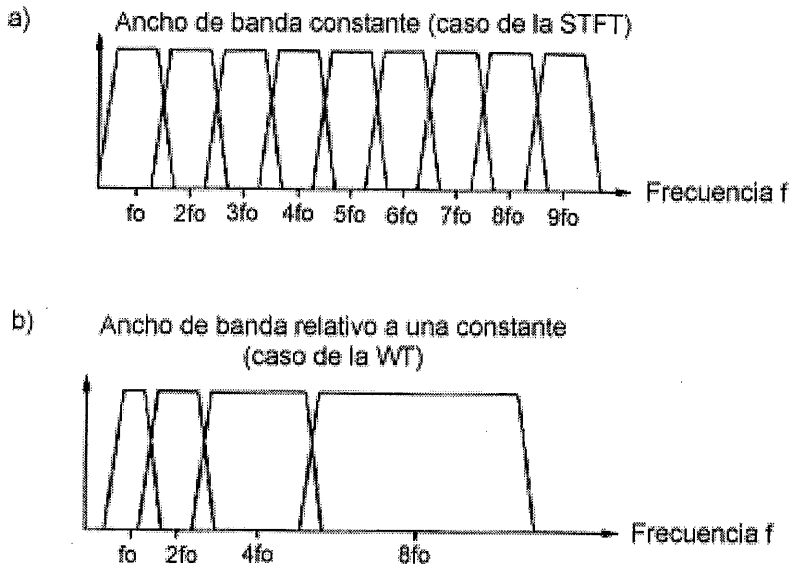

Figura 1.2: División del dominio de la frecuencia a) para la STFT (recubrimiento uniforme) y b) para la WT (recubrimiento logarítmico).

el análisis yendo a frecuencias de análisis más altas de forma de incrementar la resolución en el tiempo. Por supuesto, esta clase de análisis cuando mejor funciona es cuando la señal está formada por componentes de alta frecuencia y corta duración junto con componentes de baja frecuencia y gran duración, caso que se da frecuentemente con las señales que aparecen en la práctica.

Se obtiene una generalización del concepto de cambiar la resolución a diferentes frecuencias con los denominados *paquetes wavelet*, donde se selecciona una resolución tiempo-frecuencia arbitraria según la señal a analizar, siempre cumpliendo el principio de incertidumbre (1.5).

La Transformada Wavelet Continua (CWT) sigue exactamente las ideas expresadas más arriba añadiendo además una simplificación: todas las respuestas impulso del banco de filtros se definen como versiones *escaladas* (estiradas o comprimidas) del mismo prototipo $h(t)$, donde

$$h_a(t) = \frac{1}{\sqrt{|\,a\,|}} h(\frac{t}{a})$$

donde a es un *factor de escala* (la constante $\frac{1}{\sqrt{|a|}}$ se utiliza para normalización en energía). Esto lleva a la definición de la CWT:

$$CWT_x(\tau, a) = \frac{1}{\sqrt{\mid a \mid}} \int x(t) h^*(\frac{t-\tau}{a}) dt \tag{1.7}$$

Como se usa el mismo prototipo $h(t)$ (llamada *wavelet básica*) para todos los impulsos respuesta del filtro, no hay ninguna escala privilegiada en particular. Por lo tanto, el análisis wavelet es similar a todas las escalas. Además, esta simplificación es útil cuando se derivan propiedades matemáticas de la CWT.

Para ver de una forma más clara la conexión con la ventana modulada utilizada en la STFT, el wavelet básico $h(t)$ en (1.7) puede seleccionarse como una ventana de modulación:

$$h(t) = g(t) e^{-2j\pi f_0 t}$$

Entoces la respuesta en frecuencia del filtro de análisis satisface (1.6) con la identidad

$$a = \frac{f_0}{f}$$

Más generalmente, $h(t)$ puede ser cualquier función paso banda, y el esquema continuará funcionando. En particular, uno puede hacer caso omiso de las transformadas con valores complejos y trabajar sólo con las de valores reales.

Es importante hacer notar aquí que la frecuencia $f = af_0$ tiene poco que ver con la descrita por la STFT: en realidad, está asociada con el esquema de escalado. Como resultado de ello, esta frecuencia local, cuya definición depende del wavelet base, no está ligada ya a la modulación en frecuencia (como era el caso de la STFT), sino que está relacionada con escalas de tiempo. Esta es la razón por la que se prefiere el término "escala" frente a "frecuencia" para la CWT. El término "frecuencia" se reserva para la STFT. Nótese que se ha definido *escala* en análisis wavelet del mismo modo que se define la escala en los mapas geográficos: las escalas grandes se corresponden con señales contraidas, mientras las escalas pequeñas se corresponden con señales dilatadas.

1.5 Las nociones de escala y resolución

Vamos aquí a aclarar ambos conceptos. En primer lugar, recordemos que cuando una función $f(t)$ sufre un reescalado

$$f(t) \rightarrow f(at), a > 0$$

se contrae si $a < 1$ y se expande si $a > 1$. Ahora bien, la CWT puede escribirse también como

$$CWT_x(\tau, a) = \sqrt{a} \int x(at) h^*(\frac{t-\tau}{a}) dt \tag{1.8}$$

o bien, a través de un cambio de variable, como

$$CWT_x(\tau, a) = \sqrt{a} \int x(at) h^*(t - \frac{\tau}{a}) dt \tag{1.9}$$

La interpretación de (1.8) es que, cuando la escala crece, la respuesta impulso del filtro $h(\frac{t-\tau}{a})$ se expande en el tiempo, y toma en cuenta sólo el comportamiento a largo plazo. De forma equivalente, (1.9) indica que a medida que la escala crece, se ve una versión progresivamente más contraída de la señal a través de un filtro de longitud constante.

La de *resolución* es una noción relacionada aunque distinta. La resolución de la señal se enlaza con su contenido frecuencial. Por ejemplo, un filtrado paso bajo de una señal mantiene su escala, pero reduce su resolución.

Los cambios de escala de señales continuas en el tiempo no alteran su resolución, ya que la operación de reescalado puede invertirse. Sin embargo, en señales en tiempo discreto, incrementar la escala en el análisis conlleva un submuestreo[4], lo que reduce automáticamente la resolución. La operación consistente en ecrementar la escala (lo que implica un sobremuestreo) puede deshacerse, por lo que no afecta a la resolución. Por lo tanto, la pérdida de resolución es equivalente a una pérdida de información sobre la señal.

[4]Es decir, se toman menos muestras por unidad de tiempo que antes.

Capítulo 2

Cálculo y aplicaciones de la transformada Wavelet

Como se ha dicho en el capítulo anterior, los *wavelets* son bloques básicos de construcción de señales, al igual que las sinusoides en el caso de la Transformada de Fourier, pero que, a diferencia de éstas, no oscilan indefinidamente, sino que tienen una forma determinada. Resulta más útil que la transformada de Fourier para analizar discontinuidades o transitorios. Con la transformada wavelet (abreviadamente WT), cualquier descomposición involucra a un par de formas de onda: una se utiliza para representar las altas frecuencias y otra las bajas.

En la transformada de Fourier, cada forma de onda senoidal está caracterizada por su frecuencia de oscilación, además de su amplitud. Como la transformada wavelet utiliza bloques de construcción con una forma determinada y de duración finita, cada wavelet utilizado en la construcción de la señal está caracterizado, además de su amplitud, por su posición y su duración (es decir, su *escala*). Los respectivos coeficientes cuantifican la contribución del wavelet en ese lugar y a esa escala. Si el coeficiente de amplitud es negativo, la forma de onda se invierte.

En la figura 2.1 puede verse la descomposición de una función diente de sierra. Los wavelets utilizados para la representación de los componentes de baja frecuencia se extienden a lo largo de toda la rampa: en cambio, los de alta frecuencia capturan el salto. Los parámetros que distinguen a cada wavelet son:

- *D/S*: Detail/smooth, es decir, si la forma de onda utilizada es la que se usa para capturar detalles o para capturar la forma general de la onda (lo que equivale a decir si el wavelet utilizado es el de alta o el de baja frecuencia).

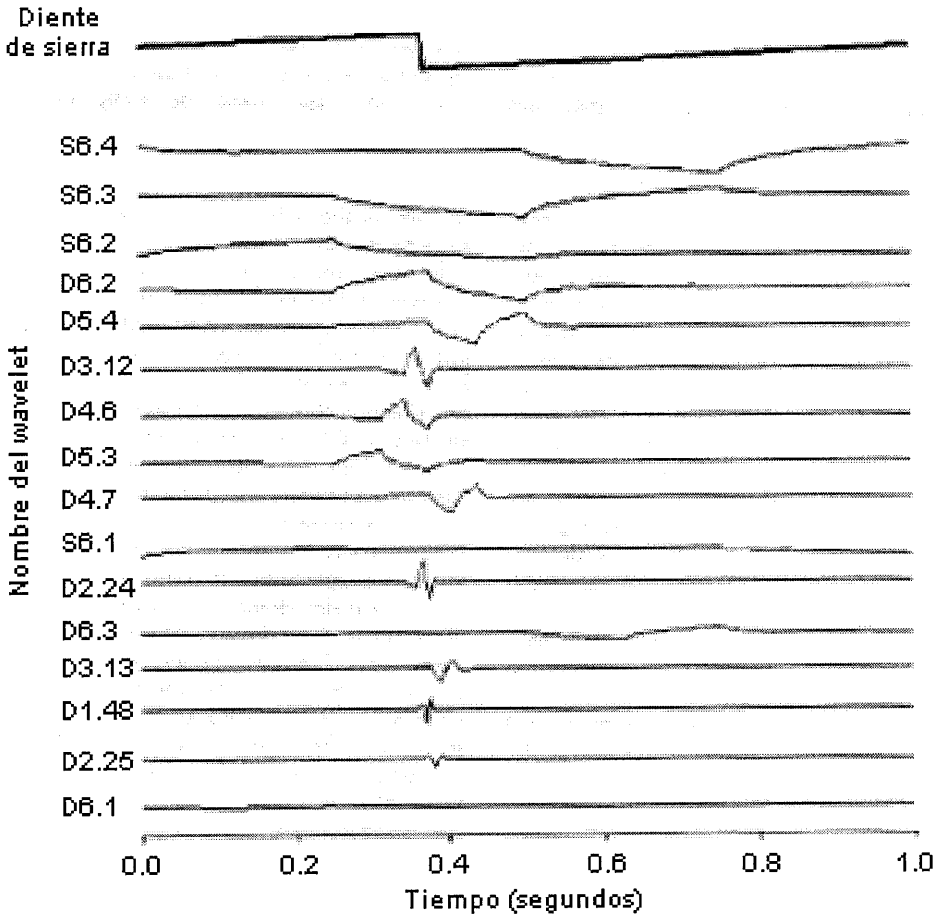

Figura 2.1: Descomposición wavelet de una señal diente de sierra en una suma de 16 wavelets de la familia "d4", que incluye tanto wavelets "gruesas" para cubrir la rampa, y wavelets "finas" para el salto en el medio. Sus nombres indican sus tipos (D de "detail", S de "smooth"), su escala y localización.

- **Escala**, es decir, la "anchura" del wavelet.
- **Amplitud**, al igual que ocurre con las sinusoides de Fourier.
- **Localización**, en una escala temporal común.

El algoritmo utilizado para la descomposición de una señal en los wavelets correspondientes se denomina *algoritmo piramidal rápido* (*fast pyramid algorithm*), desarrollado por Mallat y Meyer. Este algoritmo consta de dos partes:

Algoritmo directo Se encarga de transformar una señal en una serie de coeficientes wavelet.

Algoritmo inverso Restituye la forma de la señal original a través de los coeficientes antedichos.

A continuación describiremos someramente dichos algoritmos. Más adelante veremos su funcionamiento más en detalle.

2.1 El algoritmo piramidal rápido

El algoritmo piramidal rápido trabaja con una señal original muestreada a una tasa suficiente como para captar todos los detalles que se desean examinar. El algoritmo piramidal rápido **directo** se encarga de dividir la señal en componentes de baja y alta frecuencia, a través de filtros. Además, combina esto con operaciones de *submuestreo*[1], lo que divide en cada iteración el número de muestras por dos. En la siguiente iteración del algoritmo se trabaja con el conjunto de muestras resultante, filtrándolo y volviéndole a aplicar un submuestreo. En cada iteración del algoritmo se obtienen unos coeficientes que se utilizarán para determinar la localización y escala de los wavelets correspondientes.

El tiempo de computación se reduce de forma geométrica tras cada iteración. En efecto, dicho tiempo es proporcional al número de muestras, y si comenzamos con ocho muestras, tras una iteración tendremos cuatro, y en la iteración siguiente sólo dos. Esta característica de dividir en cada pasada el número de muestras es lo que le da nombre al algoritmo piramidal.

Para N muestras, el coste de computación es $O(n)$, frente a $O(n^2)$ de otras transformadas utilizadas con los mismos fines, o frente a $O(n.log(n))$ del

[1]El término inglés es **downsampling**, y hace referencia a descartar sistemáticamente muestras según una tasa determinada. Es decir, un submuestreo por dos (*downsampling by two*), implica descartar una muestra de cada dos.

algoritmo FFT (transformada rápida de Fourier). El tiempo de computación es $n.C$, con C dependiente de los términos que tenga el wavelet utilizado en sus filtros lineales, como veremos más adelante, cuando describamos el algoritmo en profundidad.

El algoritmo **inverso** invierte el proceso, duplicando en cada iteración el número de muestras[2] y combinando esto con el paso por filtros lineales. Se duplica el número de muestras insertando ceros en cada iteración. Posee el mismo orden de complejidad que el algoritmo directo correspondiente.

2.2 Paquetes wavelet y paquetes coseno

En realidad, la WT está en un extremo de todo un espectro de transformadas utilizadas para la representación de señales con detalles localizados en el tiempo. En el otro extremo está la STFT. En el medio aparecen otras transformadas que utilizan diferentes formas de onda como bloques de construcción básicos: los *paquetes wavelet* y los *paquetes coseno*. Ambos están localizados en el tiempo: tienen localización y duración como los wavelets. Sin embargo, oscilan muchas veces, poseyendo una frecuencia de oscilación como las sinusoides. Por lo tanto, los "paquetes" tienen cuatro parámetros: localización, duración, frecuencia y amplitud. El número de veces que oscilan no viene dado por la forma de la onda, que en el caso de los wavelets es fija, sino por la frecuencia unida a la duración. En la figura 2.2 pueden verse ejemplos de paquetes coseno y paquetes wavelet.

Debido a que comparten propiedades de frecuencia con localización en el tiempo, se las denomina "formas de onda de tiempo-frecuencia" (*time-frequency waveforms*).

Ambos paquetes son útiles en los mismos problemas que la WT, pero son mejores para representar *señales localmente oscilatorias*, como por ejemplo música digitalizada, o *imágenes localmente oscilatorias*, como texturas digitalizadas, ya que son señales con una representación más dispersa en el dominio transformado (dominio tiempo-frecuencia) que las que se obtienen aplicando sobre ellas WT o STFT. Por lo tanto, hacen falta menos paquetes básicos para representar una señal, lo que lleva a índices de compresión más altos.

Existen algoritmos rápidos para calcular las descomposiciones tiempo-frecuencia: se obtienen tiempos de $O(n.log(n))$, para n muestras, como en el caso de la FFT. Han demostrado ser de gran utilidad en la restauración de grabaciones musicales históricas, y en la compresión de huellas dactilares digitalizadas. En esta última aplicación, el estándar JPEG da una tasa de

[2]Esta operación se denomina en inglés **upsampling by two**.

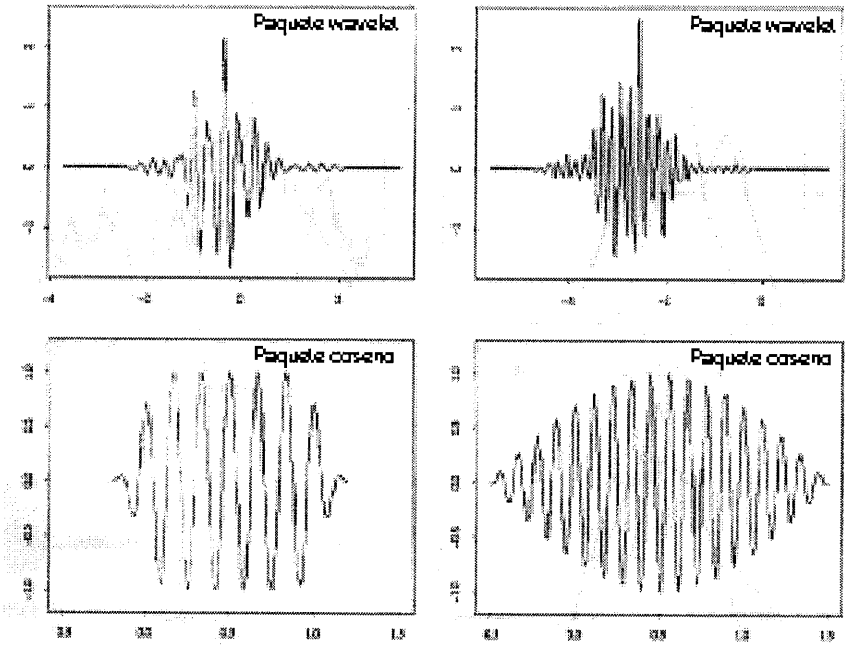

Figura 2.2: Los paquetes wavelet son similares a los wavelets, pero tienen más oscilaciones. Los paquetes coseno son funciones coseno moduladas a través de una ventana temporal (las dos figuras de abajo se obtienen a través de diferentes ventanas temporales).

compresión de 5:1, mientras que los paquetes wavelet dan 20:1. Brislawn desarrolló un estándar de compresión para el FBI, con tecnología de codificación subbanda, con resultados similares.

Otra aplicación: eliminación de ruidos en imágenes. Woog, Coifman y Johnson la utilizan para quitar ruido en fotogramas obtenidos por resonancia magnética ecoplanar.

Lo que diferencia a los paquetes coseno de los paquetes wavelet es lo siguiente. Los paquetes coseno son oscilaciones sinusoidales moduladas, mientras que los paquetes wavelet son wavelets cuyas oscilaciones persisten con una cierta frecuencia, aunque sean finitas. La transformada del coseno discreta (DST) representa la señal como suma de sinusoides. Dichas sinusoides son reales, no complejas como en Fourier. De gran utilidad en compresión de datos.

2.3 Codificación subbanda

Los métodos de codificación subbanda y sus filtros asociados están muy relacionados con las construcciones wavelet. La codificación subbanda, desarrollada en telefonía de forma independiente al desarrollo de la WT, es un mecanismo que busca comprimir la señal hablada, a través de algoritmos que dividen la señal en componentes de alta y baja frecuencia, submuestreando cada una de ellas. Como puede verse, el algoritmo piramidal es, por tanto, un caso especial de codificación subbanda. A partir de ambas subbandas, los QMF (del inglés *quadrature mirror filters*) permiten aproximar la señal original [Wic94].

Mientras que la codificación subbanda se desarrolló para comprimir la señal, el análisis wavelet busca una descomposición de ésta con propiedades adecuadas para interpretar y/o modificar los coeficientes wavelet. Debido a esto, las herramientas existentes para el cálculo de la WT discreta de una señal por métodos computacionales son más útiles para el análisis exploratorio de señales que para compresión.

Existen estudios que relacionan ambos métodos, aplicándolos de forma conjunta en la compresión de señales [RVH96].

2.4 Algunas aplicaciones

A continuación haremos un breve repaso de las aplicaciones en donde la transformada Wavelet ha demostrado su utilidad. En capítulos posteriores haremos un examen más detallado de aplicaciones concretas.

2.4.1 Compresión de datos

El algoritmo JPEG, estándar en la compresión de imágenes, toma bloques de 8x8 pixels, los convierte en serie de cosenos y guarda sus coeficientes discretos. El uso de la WT mejora notablemente los índices de compresión, al necesitar menos wavelets para representar una señal con la misma pérdida de información que la suma de cosenos utilizada.

2.4.2 Aceleración de cálculos

Los tiempos de operaciones algebraicas con matrices suelen ser típicamente de $O(n^2)$, con n el número de elementos por fila y columna de la matriz. Se pueden realizar operaciones con matrices en el dominio wavelet: utilizando métodos parecidos a los de tratamiento de imágenes, se obtiene la transformada de la matriz y se opera con ella, deshaciendo luego la transformación del resultado. Se obtiene así una buena aproximación numérica de la solución real, con un tiempo de cómputo que es una fracción del utilizado en otros métodos numéricos. Además, puede trabajarse sobre ecuaciones diferenciales con coeficientes no constantes, lo que no puede hacer la FFT: algoritmo Beylkin-Coifman-Rokhlin (BCR).

2.4.3 Eliminación del ruido

Se basa en obtener la WT de la señal afectada por el ruido y poner a cero los coeficientes de amplitud situados por debajo de un cierto umbral. Esto implica eliminar sólo el ruido, preservando los detalles de la señal original: si en un punto hay un pico de alta frecuencia y de gran amplitud, un filtro paso bajo lo eliminaría igualmente. Sin embargo, el wavelet que represente la contribución de ese pico a la señal tendrá una gran amplitud, y no será eliminado a través de este procedimiento, al superar el umbral utilizado. Luego se calcula la WT inversa de la señal. Los autores que han trabajado en este tema son D. Donoho y Johnstone, por una parte, y de forma independiente Keyachan y Picard (París). En la figura 2.3 puede verse un ejemplo.

2.4.4 Prototipado rápido de señales sísmicas

Este fue uno de los primeros campos en los que se aplicó la WT. La medición de una señal sísmica puede generar 3000 Gb de información en un par de horas. El cálculo de la WT de dicha señal permite un procesamiento rápido de la misma, además de suministrar un método de compresión muy eficaz (con tasas de hasta 100:1) para el envío de la señal via satélite. Entre los

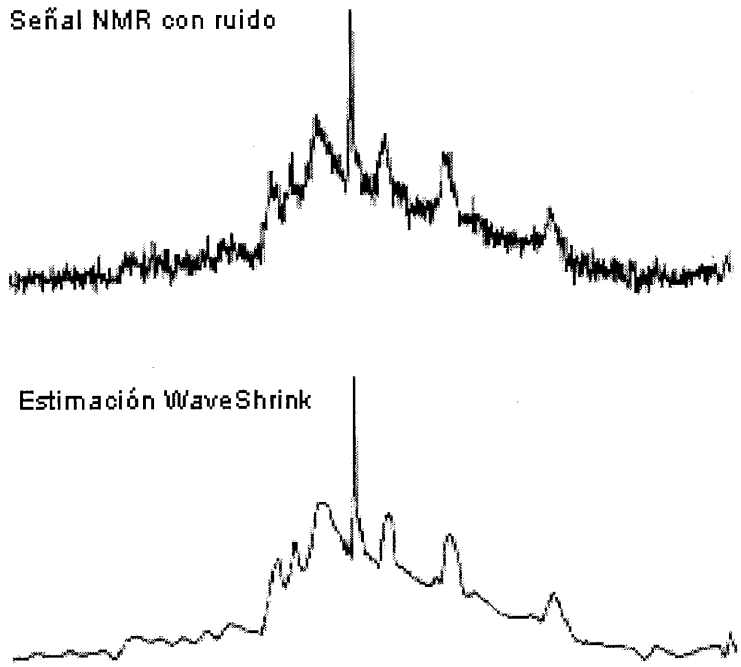

Figura 2.3: Eliminación del ruido en una señal de resonancia nuclear magnética (NMR). Puede verse que se ha eliminado el ruido preservando a la vez los picos.

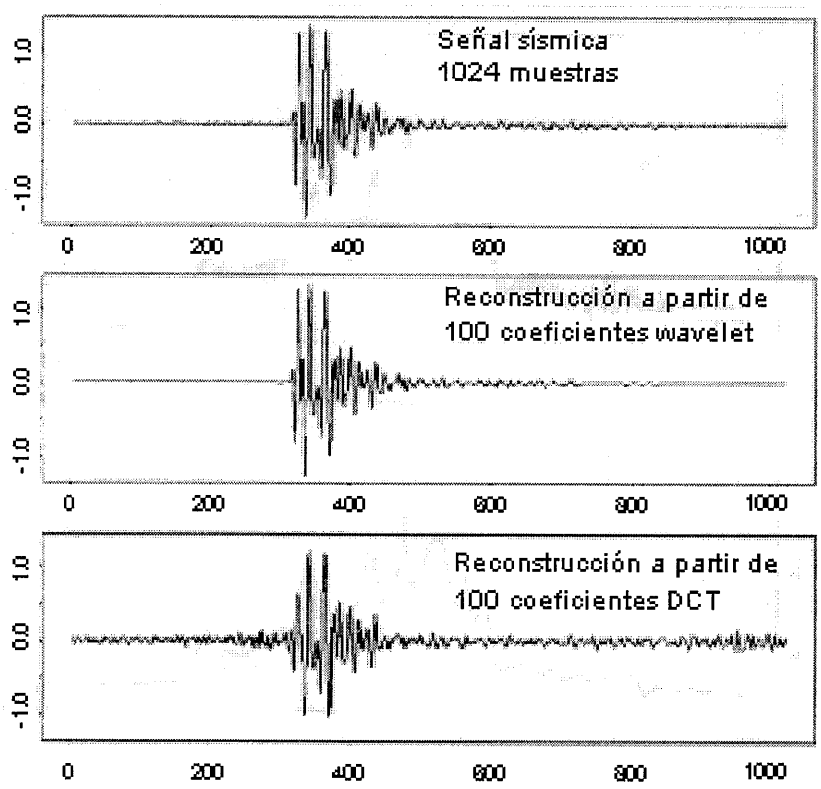

Figura 2.4: Una señal sísmica de 1024 muestras (arriba) se reconstruye a través de 100 coeficientes wavelet (en el medio). Se ha reconstruido también con 100 funciones coseno (abajo) utilizando la transformada discreta del coseno, ampliamente utilizada en compresión de señal y de imagen. La reconstrucción a través de wavelets es visiblemente mejor.

autores que han trabajado en estos temas están P. Donoho (padre de David Donoho, mencionado más arriba) y Ray Ergas. Véase la figura 2.4.

2.4.5 Astronomía

Dado que el Universo posee una estructura a diferentes escalas organizada jerárquicamente (galaxias individuales, grupos de galaxias, etc), se requiere de un modelo de visión específico para detectar, describir y clasificar cada componente de la jerarquía. Bijaoui et al. [BSRL96] han desarrollado modelos de visión multiescala para detectar estructuras a diferentes escalas, haciendo uso de un sistema de análisis automático de la imagen basado en la WT.

2.4.6 Estudio de flujo turbulento

Se ha aplicado también las técnicas basadas en wavelets para estudiar turbulencias en un fluido [FKPG96]. Existen una serie de cuestiones aún no resueltas en este campo, ya que el estudio de turbulencias para flujos muy grandes implica el estudio de sistemas en donde domina la convección no lineal. Por ello, pese a utilizarse la representación de Fourier para representar el término disipativo, no es adecuada para esta clase de sistemas.

2.4.7 Imagen

La transformación multiescala es importante para analizar la información contenida en imágenes. La teoría Wavelet brinda una base matemática sólida para comprender las propiedades de los algoritmos multiescala. Mallat [Mal96] describe las aplicaciones principales para búsqueda multirresolución, detección de bordes multiescala y discriminación de texturas.

La WT tiene aplicaciones en la síntesis además de tenerlas en el análisis. Schroder [Sch96] ha estudiado la utilidad de la WT en el manejo de escenas geométricamente complejas cuando las tasas de actualización de las imágenes son a razón de varios fotogramas por segundo.

Capítulo 3

Análisis Wavelet y procesamiento de señal

3.1 Introducción

Como se ha visto, la transformada wavelet surgió como una alternativa a la STFT y a la transformada Gabor para el análisis de señales con discontinuidades. En contraste con la STFT, que utiliza una única ventana de análisis [AR77], la WT utiliza ventanas pequeñas en frecuencias altas y ventanas grandes en frecuencias bajas. Debido a que realiza un análisis de frecuencia relativa a un ancho de banda constante, dicho análisis se denomina también *Q-constante*. La señal wavelet prototipo puede considerarse como un filtro paso banda; las propiedades *Q-constante* de los otros filtros paso banda, es decir, de los otros wavelets que aparecen en la descomposición original, se deben a que son versiones escaladas del prototipo.

En la WT, la noción de *escala* se introduce como alternativa a la de *frecuencia*, lo que lleva a una representación en un plano tiempo-escala, equivalente al plano tiempo-frecuencia de la STFT.

3.2 Análisis y síntesis wavelet

Pueden definirse los wavelets como funciones básicas, escribiendo la ecuación 1.7 como

$$CWT_x(\tau, a) = \int x(t) h^*_{a,\tau}(t) dt$$

lo que mide la "similitud" entre la señal y las funciones básicas

$$h_{a,\tau} = \frac{1}{\sqrt{a}} h\left(\frac{t-\tau}{a}\right)$$

que son versiones escaladas y trasladadas del prototipo básico $h(t)$ (véase figura 1.1d).

Las ondas senoidales básicas de la STFT son reemplazadas por señales de referencia más localizadas con parámetros de tiempo y frecuencia (o escala). De esta forma, el análisis wavelet conduce a la obtención de un conjunto de coeficientes wavelet que indican cuánto se aproxima la señal a una función básica particular. Así, cualqier señal podría representarse a través de una descomposición de wavelets, todas con la misma forma pero con diferente tamaño, amplitud y localización. Es decir, esperamos que el conjunto de wavelets $h_{a,\tau}(t)$ se comporte como una base ortogonal:

- El *análisis* se realiza calculando los productos interiores.
- La *síntesis* se realiza sumando todas las proyecciones ortogonales de la señal sobre los wavelets:

$$x(t) = c \int_{a>0} \int \mathrm{CWT}(\tau, a) h_{a,\tau}(t) \frac{da d\tau}{a^2} \tag{3.1}$$

  donde c es una constante que sólo depende de h(t).

Por supuesto, $h_{a,\tau}(t)$ no es ortogonal porque es muy redundante (está definido para a, τ, las cuales varían de forma continua. Pero, sorprendentemente, la fórmula de síntesis anterior se satisface sólo con que $h(t)$ tenga una energía finita y sea "paso banda" (lo que implica que oscila en el tiempo como una pequeña ola, de ahí el nombre de wavelets). De una forma más precisa, si se asume $h(t)$ como suficientemente regular, puede hacerse la reconstrucción sin pérdida de energía para la señal original.

Puede considererse una reconstrucción similar para la STFT, pero es menos restrictiva: sólo hace falta que la energía de la ventana sea finita.

3.3 Escalogramas

Se define el espectrograma como el módulo cuadrado de la STFT. El espectrograma suministra la distribución de la energía de la señal en el plano tiempo-frecuencia. Como la CWT se comporta como una descomposición en bases ortonormales, es *isométrica*, es decir, preserva la energía de la señal original:

$$\int \int |\mathrm{CWT}(\tau, a)|^2 \frac{d\tau da}{a^2} = Ex$$

donde

$$Ex = \int |x(t)|^2 dt$$

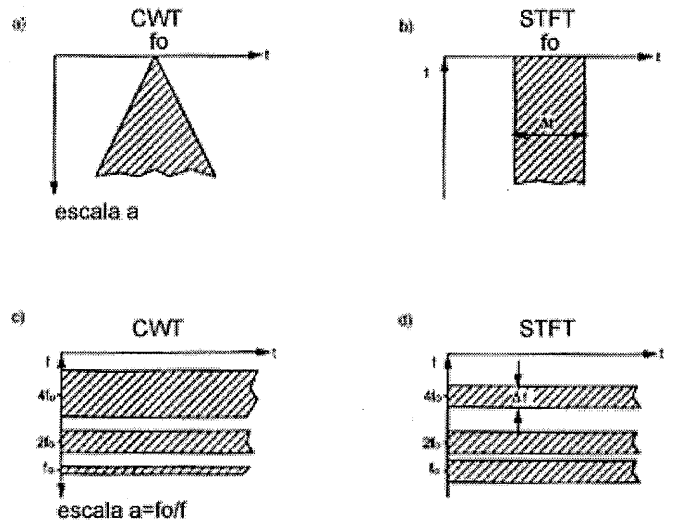

Figura 3.1: Regiones de influencia de un pulso de Dirac en $t = t_0$ a) para la CWT y b) para la STFT; así como de tres sinusoides de frecuencias f_0, $2f_0$ y $4f_0$ para c) la CWT y d) la STFT.

es la energía de la señal $x(t)$. Así se define el *escalograma* o *espectrograma wavelet*: el módulo cuadrado de la CWT. Es la distribución de la energía de la señal en el plano tiempo-escala, expresada en potencia por unidad de frecuencia, como el espectrograma. Pero la energía aquí se distribuye con diferente resolución, de acuerdo con la figura 1.1b.

La figura 3.1 muestra que la influencia del comportamiento de la señal alrededor de $t = t_0$ en el análisis se limita a un cono en el plano tiempo-escala: está muy localizado alrededor de t_0 para escalas pequeñas. En la STFT es tan ancha como la ventana de análisis a todas las frecuencias.

Similarmente, como el análisis tiempo-escala es logarítmico en frecuencia, el área de influencia de una frecuencia pura f_0 en la señal crece con f_0 en el escalograma, mientras permanece constante en el espectrograma.

Aunque tanto el escalograma como el espectrograma producen una representación más o menos interpretable de la señal, tienen sus desventajas. En general, no pueden invertirse para reconstruir la señal, ya que falta la información de fase. Además, como son funciones bilineares de la señal analizada, aparecen términos cruzados como interferencias en el diagrama.

Además, se ha visto que la representación de la fase en el caso de los wavelets revela ráfagas locales aisladas mejor que el escalograma, que muestra la potencia.

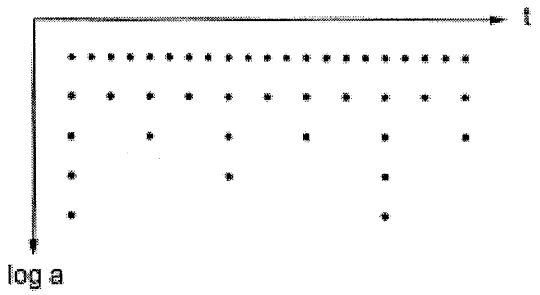

Figura 3.2: Rejilla de muestreado diádica en el plano tiempo-escala

El trabajo realizado por diferentes autores para unir los espectrogramas, escalogramas y la distribución de Wigner dentro de una clase común de representaciones de energía indica que hay fuertes enlaces entre la WT y la distribución Wigner-Ville. Incluso es posible ir del espectrograma de una señal dada a su escalograma, a través de dicha distribución. Esta propiedad puede ayudar a decidir si debemos elegir el análisis tiempo-escala o tiempo-frecuencia para un problema dado.

3.4 Tramas wavelet y bases ortonormales

3.4.1 Discretización de parámetros tiempo-escala

Se ha visto que los wavelets continuos se comportan en el análisis y síntesis wavelet como una base ortonormal. La pregunta es: si se discretizan los parámetros de tiempo-escala (a, τ), ¿puede obtenerse una base ortonormal *real*? La respuesta depende de la elección del wavelet básico $h(t)$.

Hay una forma natural de discretizar a, τ: como dos escalas $a_0 < a_1$ corresponden *grosso modo* a dos frecuencias $f_0 > f_1$, los coeficientes wavelet a escala a_1 pueden submuestrearse en una proporción f_0/f_1 de los coeficientes a escala a_0, de acuerdo con la regla de Nyquist. Así, pueden dicretizarse los parámetros de tiempo-escala en el entramado mostrado en la figura 3.1. Así, se tiene $a = a_0^j$ y $b = ka^jT$, donde j, k son enteros. Los wavelets correspondientes serán:

$$h_{j,k}(t) = a_0^{-j/2} h(a_0^{-j}t - kT) \tag{3.2}$$

lo que lleva a los coeficientes wavelet

$$c_{jk} = \int x(t) h^*_{j,k}(t) dt \tag{3.3}$$

El problema de reconstrucción se reduce así a encontrar a_0, T y $h(t)$ de forma que

$$x(t) \approx c \sum_j \sum_k c_{j,k} h_{j,k}(t) \tag{3.4}$$

donde c es una constante que no depende de la señal (comparar esto con la ecuación 3.1).

Si a_0 es cercano a 1 y T es lo suficientemente pequeño, la ecuación 3.4 se acerca mucho a la ecuación 3.1, y se consigue la reconstrucción sin condiciones restrictivas para $h(t)$. Del otro lado, si el muestreo es disperso, por ejemplo, si la computación se hace octava a octava ($a_0 = 2$), sólo se consigue una base ortonormal real para elecciones de $h(t)$ muy especiales. En otras palabras, si se realiza un muestreo de forma que sólo se obtengan las muestras "justas", sin información redundante, las restricciones a $h(t)$ son fuertes.

3.4.2 Tramas wavelet

La teoría de tramas wavelet suministra un marco de trabajo que cubre ambas situaciones. Nos permite obtener un balance entre:

- **Redundancia** (es decir, densidad de muestreo).
- **Restricciones** en $h(t)$ de forma que el esquema de reconstrucción descrito en la ecuación 3.4 funcione.

Si hay una gran redundancia (sobremuestreo), se ponen pocas restricciones a la ecuación 3.2. Si la redundancia es pequeña (cercana al muestreo "crítico"), las funciones básicas utilizables se restringen.

La suposición que realiza Daubechies [Dau90] es que el operador lineal que relaciona a $x(t)$ con $c_{j,k}$ está acotado. Se denomina entonces *trama* a la familia de funciones wavelet, y se cumple que la energía de los coeficientes $c_{j,k}$ relativos a la señal se ajusta entre dos "cotas de trama" positivas, A y B:

$$A.Ex \leq \sum_{j,k} |c_{j,k}|^2 \leq B.Ex$$

con Ex igual a la energia de la señal $x(t)$.

Estos límites pueden calcularse a partir de a_0, T y $h(t)$ utilizando las fórmulas de Daubechies [Dau90]. Además, regulan la corrección de la señal reconstruida por la ec. 3.4:

$$x(t) \approx \frac{2}{A+B} \sum_j \sum_k c_{j,k} h_{j,k}(t)$$

con una relación señal/ruido mayor que

$$\frac{(B/A)+1}{(B/A)-1)}$$

Cuanto más cerce estén A y B, más correcta será la reconstrucción. Si se cumple que $A = B$, los wavelets se comportan exactamente como una base ortonormal, aunque pueden no ser linealmente independientes.

La reconstrucción puede hacerse exacta en el caso general si uno utiliza diferentes funciones de síntesis $h'_{j,k}(t)$, lo que constituye la llamada *trama dual* de $h_{j,k}(t)$.

3.4.3 Bases de wavelets ortogonales

Si una trama es lo suficientemente estrecha como para que todos los wavelets $h_{j,k}(t)$ sean necesarios para reconstruir la señal, entonces forman una base ortonormal del espacio de señales con energía finita. Recordemos que "ortonormal" significa que

$$\int h_{j,k}(t).h'_{j,k}(t)dt = \begin{cases} 1 & \text{si } j = j' \text{ y } k = k' \\ 0 & \text{en otro caso} \end{cases}$$

Así, una señal arbitraria puede representarse exactamente como una suma ponderada de funciones básicas:

$$x(t) = \sum_{j,k} c_{j,k} h_{j,k}(t)$$

Además, en este caso las funciones básicas $h_{j,k}(t)$ se obtienen a partir de una única función prototipo $h(t)$ por escalamientos y desplazamientos. Existen funciones $h(t)$ que se comportan de esta manera, como se verá más adelante.

En contraste, con la STFT, de acuerdo con el teorema de Balian-Low [Dau90], es imposible tener bases ortonormales con funciones bien localizadas en el tiempo y en la frecuencia, es decir, para las que $\Delta t.\Delta f$ es un número finito. Recientemente, el esquema ortonormal de wavelets se ha extendido a funciones de síntesis $h'_{j.k}(t) \neq h_{j,k}(t)$, lo que ha llevado a las denominadas *bases biortogonales* de wavelets.

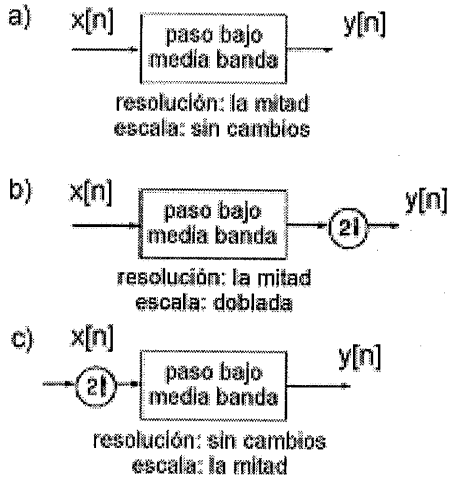

Figura 3.3: Cambios en la resolución y en la escala en tiempo discreto (por un factor de 2).

3.5 El caso de tiempo discreto

En el caso de tiempo discreto, se desarrollaron dos métodos de forma independiente a finales de los setenta y principios de los ochenta. Ambos llevaron a la DWT. Dichos métodos son:

- Codificación subbanda.
- Codificación piramidal o análisis de señal multirresolución.

Ambos métodos fueron propuestos para codificación, por lo que la noción de muestreo crítico (mínimo número de muestras) tiene importancia. En primer lugar se estudiará el segundo de estos métodos. Recordamos además que el parámetro de escala "grande" indica que wavelets dilatados toman vistas "globales" de una señal submuestreada, mientras que para pequeñas escalas, los wavelets contraidos analizan pequeños detalles en la señal. Véase la figura 3.3.

3.5.1 Pirámide multirresolución

Dada una secuencia $x[n]$, con $n \in \mathcal{Z}$, puede derivarse una señal de más baja resolución filtrándola a través de un filtro de paso bajo de media banda, con una respuesta al impulso $g[n]$. Siguiendo la regla de Nyquist, se puede

submuestrear por dos, lo que dobla la escala en el análisis. Esto lleva a una señal $y[n]$ dada por

$$y[n] = \sum_{k=-\infty}^{\infty} g[n]x[2n-k]$$

El cambio en la resolución se obtiene a través del filtrado de paso bajo, ya que permite perder detalles en alta frecuencia.

A partir de esta versión de paso bajo y submuestreada de $x[n]$, se intentará encontrar una aproximación $a[n]$ a la señal original. Esto se consigue sobremuestreando $y[n]$ por dos [1] (insertando un cero entre cada muestra), con lo que se obtiene una señal a la escala original. Así,

$$y'[2n] = y[n],\ y'[2n+1] = 0$$

Luego $y'[n]$ se interpola a través de un filtro con respuesta al impulso $g'[n]$ para obtener la aproximación $a[n]$:

$$a[n] = \sum_{k=-\infty}^{\infty} g'[k]y'[n-k]$$

Si $g[n]$ y $g'[n]$ fueran filtros de media banda perfectos (es decir, teniendo una frecuencia paso banda igual a 1 sobre el rango de frecuencia normalizado $-\pi/2$, $\pi/2$ e igual a cero en otra parte), la transformada Fourier de $a[n]$ sería igual a la transformada Fourier de $x[n]$ sobre el rango de frecuencias $(-\pi/2, \pi/2)$, y cero fuera de él. Es decir, $a[n]$ sería una aproximación paso bajo de media banda de $x[n]$ perfecta.

En el caso general, la diferencia entre ambos será distinta de cero. Llamemos $d[n]$ a dicha diferencia:

$$d[n] = x[n] - a[n]$$

$x[n]$ puede reconstruirse a partir de $d[n]$ y $a[n]$ (véase la figura 3.7). Sin embargo, habría cierta redundancia, porque se está mapeando una señal con tasa de muestreo f_s en dos señales $d[n]$ e $y[n]$ con tasas de muestreo f_s y $f_s/2$ respectivamente.

Si el filtro paso banda fuera perfecto, está claro que $d[n]$ contendría exactamente las frecuencias sobre $\pi/2$ de $x[n]$, por lo que $d[n]$ podría submuestrearse por dos sin pérdida de información.

La separación de la señal original $x[n]$ en una aproximación gruesa a $a[n]$ y unos detalles adicionales contenidos en $d[n]$ es conceptualmente importante.

[1] Los términos *submuestrear* y *sobremuestrear* utilizados en este capítulo son traducción de los términos en inglés *downsampling* y *upsampling*.

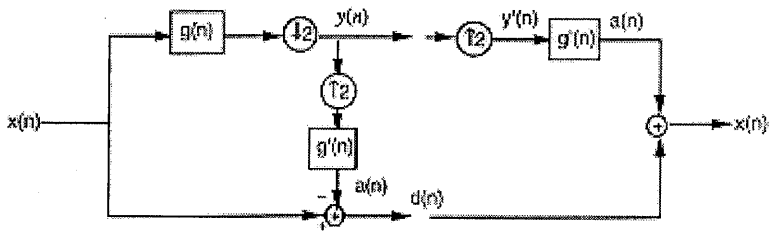

Figura 3.4: Esquema piramidal

Debido al cambio de resolución implicado (un filtrado paso bajo seguido de un submuestreo por dos que produce una señal con la mitad de resolución y el doble de escala que la señal original), este método se llama *análisis de señal multirresolución*, y se utiliza en procesamiento de imagen.

Este esquema puede iterarse sobre $y[n]$, creando una jerarquía de señales de más baja resolución a escalas más pequeñas. Debido a esta jerarquía y al hecho de que las señales se van acortando sucesivamente (o bien las imágenes sobre las que se aplica este método se van haciendo más pequeñas), estos esquemas se denominan *pirámides de señal* (o de imagen).

3.5.2 Esquemas de codificación subbanda

Como se ha dicho, una parte de la descomposición piramidal lleva a una señal de la mitad de la tasa original y de baja resolución y a una señal diferencia con la misma tasa, lo que incrementa en un 50 por ciento el número de muestras. Este sobremuestreo puede evitarse si los filtros $g[n]$ y $g'[n]$ cumplen determinadas condiciones.

Se verá a continuación otro esquema que no posee redundancia. Se trata del esquema de codificación subbanda, utilizado en compresión de voz.

La aproximación de paso bajo, submuestreada, se obtiene exactamente como se ha visto más arriba, pero, en lugar de una señal de diferencia, se computa el "detalle añadido" como una versión de $x[n]$ filtrada en paso alto con un filtro de respuesta al impulso $h[n]$, seguido de un submuestreo por dos.

Intuitivamente, está claro que ese "detalle añadido" a la aproximación paso bajo tiene que ser una señal paso alto, y es obvio que si $g[n]$ es un filtro paso bajo de media banda ideal, entonces un filtro ideal de paso alto y media banda $h[n]$ llevará a una representación perfecta de la señal en dos versiones submuestreadas.

Esto es exactamente un paso de una descomposición wavelet utilizando

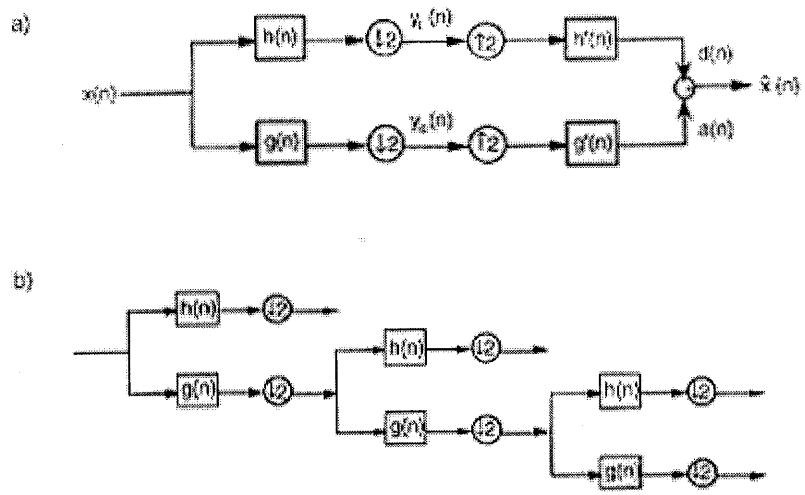

Figura 3.5: Esquema de codificación subbanda. a) Se calculan dos aproximaciones submuestreadas, una de bajas y otra de altas frecuencias. La señal reconstruida se obtiene reinterpolando las aproximaciones y sumándolas. b) Arbol de banco de filtros de la DWT implementada con filtros de tiempo-discreto y submuestreo por dos. La resolución en frecuencia aparece en la figura 1.2b.

filtros $sen(x)/x$, ya que la señal original se mapea en una aproximación paso bajo al doble de la escala y en otra paso alto también a doble escala (recordamos aquí que "doble escala" significa mirar la señal desde "más lejos").

En particular, utilizando estos filtros ideales, la versión discreta es idéntica a la Transformada Wavelet Continua (CWT).

Lo mejor de todo es que no es necesario utilizar filtros ideales para que $x[n]$ pueda recuperarse a partir de las dos versiones filtradas y submuestreadas, que llamaremos $y_0[n]$ e $y_1[n]$, aunque, como veremos, tampoco vale cualquier filtro. Para recuperar dicha señal, lo que se hace es sobremuestrearla y filtrarla a través de $g'[n]$ y $h'[n]$ respectivamente, y luego se suman (véase la figura 3.5a). Ahora, a diferencia del caso piramidal, la señal reconstruida (a la que llamaremos $\hat{x}[n]$) no es idéntica a $x[n]$, al menos que los filtros, sin ser ideales, cumplan determinadas restricciones. Los que lo hacen se dice que tienen la propiedad de la "reconstrucción perfecta".

El caso más fácil a analizar aparece cuando los filtros de análisis y síntesis de la figura 3.5a son idénticos (dentro de la inversión en el tiempo) y se obtiene una reconstrucción perfecta ($\hat{x}[n] = x[n]$, con un posible desplazamiento). Entonces puede demostrarse que el análisis/ síntesis subbanda corresponde a

una descomposición en una base ortogonal, seguida de una reconstrucción que se obtiene sumando las proyecciones ortogonales. Se asumirán en lo sucesivo filtros FIR. Esto hace que los filtros paso bajo y paso alto se relacionen a través de

$$h[L-1-n] = (-1)^n g[n] \tag{3.5}$$

donde L es el largo del filtro (que debe ser par). Nótese que la modulación por $(-1)^n$ transforma el filtro paso alto en uno paso bajo.

Ahora, el banco de filtros de la figura 3.5a, que computa convoluciones seguidas de un submuestreo por dos, evalúa el producto interior de la secuencia $x[n]$ y las secuencias $g[-n+2k]$, $h[-n+2L]$ (la inversión en el tiempo viene de la convolución, que invierte una de las secuencias). Así,

$$y_0[k] = \sum_n x[n]g[-n+2k]$$

$$y_1[k] = \sum_n x[n]h[-n+2k]$$

Como la respuesta al impulso de los filtros forman un conjunto ortonormal, se deduce que $x[n]$ es una suma ponderada de las respuestas ortogonales al impulso:

$$x[n] = \sum_{k=-\infty}^{\infty} (y_0[k]g[-n+2k] + y_1[k]h[-n+2k]) \tag{3.6}$$

donde los pesos son los productos interiores de la señal con las respuestas al impulso. Esta es, como se ha visto, la expansión estándar de una señal en una base ortonormal.

De la ecuación 3.5 y la 3.6 también queda claro que los filtros de síntesis son idénticos a los de análisis, dentro de la inversión en el tiempo.

Estos bancos de filtros con reconstrucción ortogonal perfecta han sido estudiados en la literatura de procesamiento digital de la señal, y a la descomposición ortogonal descrita se la llama banco de filtros "paraunitario" o "sin pérdida". Entre sus propiedades figura que pueden extenderse a más de dos canales.

Los filtros *biortogonales* son bancos de filtrado con una reconstrucción perfecta más generales, y aparecen de forma natural al implementar la CWT.

Hasta el momento se ha asumido un procesamiento lineal. Si el procesamiento no es lineal (como sucede en la cuantificación), la naturaleza "sobremuestreada" del esquema de pirámide descrito antes puede llevar a una mayor robustez.

3.6 La Transformada Wavelet Discreta

Para la descomposición se han usado filtros ortogonales (respecto a desplazamientos pares). El proceso visto puede iterarse tanto en una como en las dos subsecuencias. En particular, para conseguir una mejor resolución en frecuencia a frecuencias más bajas, como lo que se obtiene con la Transformada Wavelet Continua, se itera el esquema sólo en la banda baja. Si $g[n]$ es un buen filtro paso bajo, por la ecuación 3.5 $h[n]$ es un buen filtro paso alto. Una iteración del esquema en la primera banda baja crea una nueva banda baja que se corresponde con la cuarta parte inferior del espectro de frecuencias (ver la figura 1.1b). Cada iteración posterior divide el ancho de la banda baja (incrementando la resolución en frecuencia por dos) pero, debido al submuestreo por dos, la resolución en el tiempo también se divide por dos. En cada iteración, la porción más alta de la banda se corresponde con la diferencia entre la porción de banda baja anterior y la actual, esto es, un paso banda (esto equivale al diagrama de bloques de la figura 3.5b, y el resultado es lo que se ve en la figura 1.2b).

Es importante destacar que este algoritmo es de baja complejidad. De hecho, es sorprendente constatar que, independientemente de la profundidad del árbol de la figura 3.5b, la complejidad es lineal con el número de muestras de entrada, con un factor constante que depende de la longitud del filtro, L. En efecto, si dicho factor es C_0, el primer nivel del algoritmo requerirá típicamente C_0 operaciones por muestra. El segundo nivel, al tener la mitad de muestras, requerirá $\frac{C_0 * m}{2}$ operaciones (con m el número de muestras), y así sucesivamente. De esta forma, el coste total es

$$C_0 \left(m + \frac{m}{2} + \frac{m}{4} + ... \right) < C_0 2m$$

Esto demuestra la eficiencia del algoritmo de la DWT y muestra que mantiene el mismo orden de complejidad de forma independiente del número de octavas que se compute. Una posible desventaja es que el retardo asociado con dicho banco de filtros iterativo crece exponencialmente con el número de niveles.

3.6.1 Filtros iterados y regularidad

Hay una diferencia importante entre el esquema discreto que se ha visto y la CWT. En el caso de tiempo discreto, el rol del wavelet lo interpreta el filtro de paso alto $h[n]$ y la cascada de filtros paso bajo submuestreados seguidos de un filtro paso alto. Como se ha visto, dicha cascada es equivalente a un filtro paso banda. Estos filtros, que se corresponden grosso modo con filtros de octavas,

no son versiones exactamente escaladas unas respecto de otras, como sucede en la CWT. En particular, como estamos en tiempo discreto, el escalado no se define tan fácilmente, ya que involucra interpolación y expansión en el tiempo.

Sin embargo, bajo ciertas condiciones, el sistema discreto converge (después de un cierto número de iteraciones) a un sistema donde los filtros subsecuentes son versiones escaladas unos de otros. Actualmente, esta convergencia es la base para la construcción de bases wavelet soportadas de forma compacta y de tiempo continuo [Dau88].

Intentaremos encontrar el filtro equivalente que se corresponda con el camino de abajo en la figura 3.5b, es decir, el filtro paso bajo iterado. Es conveniente utilizar transformadas Z de los filtros, como por ejemplo

$$G(z) = \sum_n g[n] z^{-n}$$

en lo sucesivo. Se ve fácilmente que submuestrear por dos seguido de un filtrado por $G(z)$ es equivalente a filtrar por $G(z^2)$ seguido del submuestreo, ya que z^2 inserta ceros entre las muestras de la respuesta al impulso, que son borrados por el submuestreo posterior. Así, los primeros dos pasos del filtrado paso bajo pueden reemplazarse por un filtro con transformada Z $G(z).G(z^2)$, seguido de un submuestreo por 4. En general, el filtro equivalente para la iteración i del filtrado paso bajo y submuestreo por dos tiene la forma

$$G^i(z) = \prod_{L=0}^{i-1} G(z^{2^L})$$

y sea su respuesta al impulso $g^i[n]$.

A medida que i tiende a infinito, el filtro se hace infinitamente largo. En su lugar, consideremos una función $f^i(x)$ constante a trozos ("piecewise") en intervalos de longitud $1/2^i$ y con valor $2^{i/2} g^i[n]$ en el intervalo $[n/2^i, (n+1)/2^i]$. Puede verse que la función está definida en $[0, L-1]$, con L la longitud del filtro $g[n]$.

Ahora bien, con i tendiendo a infinito, $f^i(x)$ puede converger a una función continua $g_c(x)$, o a una función con un número finito de discontinuidades, o incluso a una función fractal, o no converger.

3.6.2 Funciones de escalado y wavelets obtenidos a partir de filtros iterados

Recordemos que $g_c(x)$ es la función final a la que converge $f^i(x)$. Dado que es el producto de filtros paso bajo, la función final también es paso bajo y

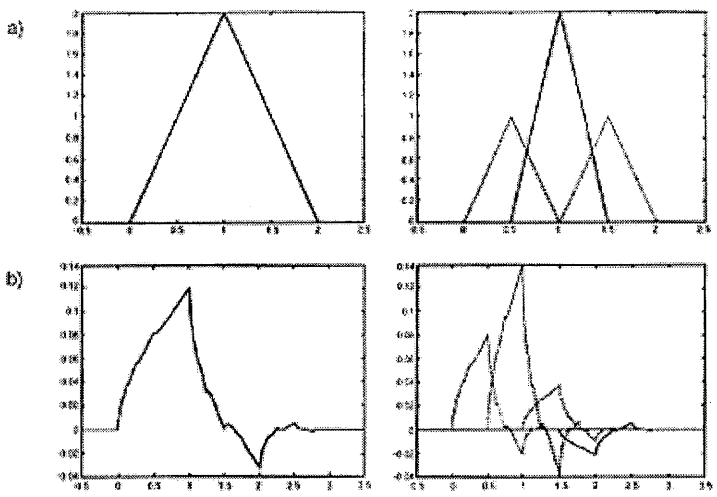

Figura 3.6: Funciones de escalado que satisfacen ecuaciones diferenciales de dos escalas. a) La función "sombrero". b) El wavelet D4 obtenido por Daubechies a partir del filtro regular "4-tap" (utilizado en la figura 2.1 para descomponer una señal diente de sierra).

se la denomina "función de escalado", porque permite ir de una escala fina a otra más gruesa. Debido al producto de la ecuación 2.16 del que se deriva la escala, $g_c(x)$ satisface la siguiente ecuación en diferencias [Dau90]:

$$g_c(x) = \sum_{n=-\infty}^{\infty} g[n] g_c[2x - n] \tag{3.7}$$

Las figuras 3.6a y 3.6b muestran dos ejemplos de funciones de escalado.

A partir de la figura 3.5b puede verse que un filtro paso banda se obtiene de la misma manera, excepto por un filtro paso alto final. Así, el wavelet $h_c(x)$ se obtiene como

$$h_c(x) = \sum_{n=-\infty}^{\infty} h[n] g_c[2x - n] \tag{3.8}$$

Puede verse [Dau88] que el conjunto $h_c(2^{-i}x - k)$, con $i, k \in \mathcal{Z}$ forma una base ortonormal para el conjunto de funciones integrables al cuadrado.

La figura 3.7 muestra dos escalas y desplazamientos del wavelet de Daubechies D4, obtenido a través del filtro regular "4-tap". [Dau88].

La figura 3.8 muestra un wavelet ortogonal basado en un filtro regular de longitud 18. Es mucho más suave que la anterior (de hecho, posee tres

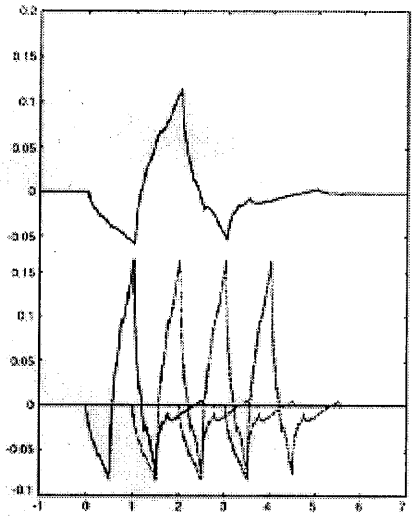

Figura 3.7: Dos escalas y desplazamientos del wavelet D4. Este conjunto de funciones es ortogonal.

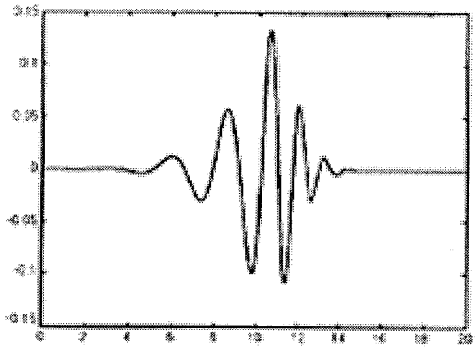

Figura 3.8: Wavelet ortonormal generado a partir de un filtro regular de tamaño 18 [Dau88], en el dominio del tiempo.

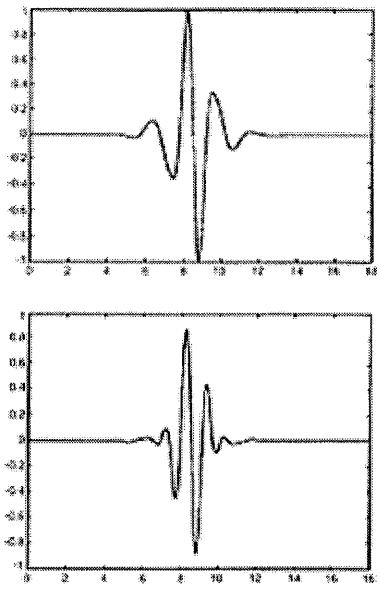

Figura 3.9: Wavelets biortogonales. a) Wavelet de análisis en el dominio del tiempo. b) Wavelet de síntesis en el dominio del tiempo.

derivadas continuas). La figura 3.9 muestra un conjunto biortogonal de wavelets de fase lineal, en donde el wavelet de análisis es ortogonal al de síntesis [VH90].

Se cumple también que pueden usarse los conjuntos ortonormales de funciones de escalado y wavelets para generar bancos de filtros de reconstrucción perfecta. Existen además extensiones del concepto de wavelets para dimensiones múltiples, lo que es útil, por ejemplo, en codificación de imágenes.

3.6.3 Filtros escalados regulares

La estructura de cálculos en la DWT y en un banco de filtros por octavas es idéntica. Sin embargo, aparte de las distintas interpretaciones que reciben ambos, la principal diferencia radica en el diseño del filtro. Los filtros wavelet se eligen de modo que sean "regulares". Esto significa que la función constante a tramos asociada con la "secuencia wavelet" discreta $h_j[n]$ de transformada Z $G^j(z)H(z^{2j})$ converge a una función límite regular $h_c(x)$. De forma equivalente, la función continua a trozos asociada con la secuencia de "escalado" discreta $g[n]$ de transformada Z $G^j(z)$ converge a una función límite regular $g_c(x)$. Por "regular" se entiende que al menos sea continua, o mejor aún, diferenciable una o dos veces. El orden de regularidad es el núme-

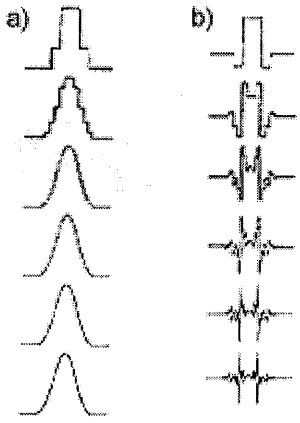

Figura 3.10: El filtro paso bajo iterado $g_j[n]$ con $g[n] = (1, 3, 3, 1)$ (izquierda) converge a una función regular. El filtro paso bajo iterado $g_j[n]$ con $g[n] = (-1, 3, 3, -1)$ (derecha) converge a una función fractal.

ro de veces que es continuamente diferenciable. La figura 3.10 muestra a la derecha una función $g_c(x)$ diferenciable al menos tres veces, y a la izquierda una que diverge con comportamiento fractal.

Cuanto más regular sea la función límite, más rápida es la convergencia a ese límite [Rio91]. Como en la práctica dicha convergencia es muy rápida, esto justifica el estudio de $h_c(x)$, que se alcanza tras unas pocas octavas de descomposición logarítmica. Dado que un error en un coeficiente wavelet (debido, por ejemplo, a la cuantificación) se convierte, después de la reconstrucción, en un error proporcional a un wavelet discreto $h_j[n]$, la regularidad es una propiedad interesante de cara a evitar distorsiones.

De las ecuaciones 3.5, 3.7 y 3.8 se ve que el conocimiento de $g[n]$ es suficiente para determinar el límite $h_c(x)$. Se han desarrollado métodos para estimar el orden de regularidad de $h_c(x)$ a partir de los coeficientes $g[n]$.

Aún no se conoce el mínimo orden de regularidad para un buen rendimiento en la codificación a través de DWT: es un campo actual de investigación.

Parte II

Aplicaciones de la Transformada Wavelet en el análisis de señales temporales

Capítulo 4

Filtrado de señales de voz con "Wavelet Shrinkage"

Publicado en las Actas IV Jornadas de Informática, organizadas por el Grupo de Arquitectura y Concurrencia (GAC), Departamento de Ingeniería Telemática, Universidad de Las Palmas de Gran Canaria. Depósito Legal GC-582-1998, ISBN 84-87526-61-6. Julio 1998.

4.1 Introducción

La técnica conocida como "wavelet shrinkage" ([DJ92], [Don92b], [Don93]) hace referencia a la reconstrucción de una señal afectada por ruido a través del siguiente procedimiento:

1. Obtener la transformada wavelet de la señal.
2. Minimizar los coeficientes cercanos a cero de la transformada.
3. Obtener la transformada inversa de la señal.

La ventaja de este procedimiento respecto de un filtrado por bandas de frecuencia reside en que se obtiene una señal casi libre de ruido, sin modificar apenas las características de la señal (presencia de picos de alta frecuencia, etcétera). Este resultado es muy distinto al que se obtiene mediante los métodos tradicionales de suavizado, que sólo consiguen eliminar el ruido a costa de suavizar también los componentes de la señal ([RV91]). Entre estos métodos podemos citar la utilización de splines con elección adaptativa del parámetro de tensión [EC98]. Otros métodos, como la estimación en Series de

Fourier, respetan las características de la señal, pero no eliminan realmente el ruido [Don93].

Supongamos que estamos interesados en filtrar una función $f(t)$ compuesta de $n = 2^{j+1}$ muestras de la forma $y_i = f(t_i) + \sigma z_i$, con $i = 1, ...n$, con los valores de t_i equidistantes y con z_i ruido blanco. Donoho y Johnstone [DJ92] han propuesto un método que consta de tres partes para recuperar $f(t)$:

1. Aplicar el filtrado wavelet piramidal propuesto por Cohen, Daubechies, Jawerth y Vial [CDJV92] al conjunto de datos, con lo que se obtiene un conjunto de coeficientes wavelet $w_{j,k}$, con $j = j_0, ..., J$ las diferentes escalas de resolución, y $k = 0, ..., 2^j - 1$ el número de muestras en cada escala.

2. Aplicar una contracción no lineal a los coeficientes wavelet obtenidos

$$\nu(w) = sign(w)(|w| - t)_+ \tag{4.1}$$

utilizando para ello un umbral u igual a

$$u = \frac{\sqrt{2log(n)}\sigma}{\sqrt{n}} \tag{4.2}$$

lo que lleva a unos nuevos coeficientes $\hat{w}_{j,k}$.

3. Poner a cero todos los coeficientes $\hat{w}_{j,k}$ para $j > J$ (lo que equivale a eliminar el residuo después de calcular la transformada wavelet para los J escalas superiores), y realizar la transformación inversa, lo que permite obtener un conjunto de muestras $\hat{f}(t)$.

Este método contrae los coeficientes wavelet cercanos a cero de forma empírica. Lo que se consigue con la contracción no lineal descrita en el paso 2 es lo siguiente:

- Si el valor absoluto del coeficiente w es mayor que el umbral u, se resta ese umbral del coeficiente.
- Si el valor absoluto del coeficiente es menor o igual al umbral, ese coeficiente se pone a cero.

Esta contracción es denominada por Donoho y Johnstone como "contracción suave" (*soft thresholding*) Alternativamente, también se propone una "contracción firme" (*hard thresholding*) consistente en poner a cero todos los coeficientes por debajo del umbral, dejando el resto inalterados.

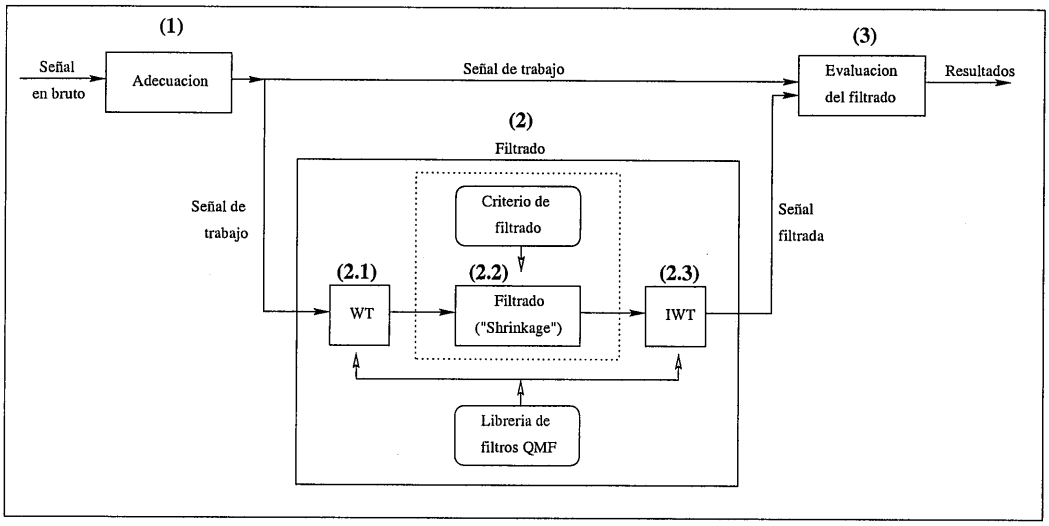

Figura 4.1: Diagrama de bloques del método de filtrado en el dominio wavelet.

De la lectura del artículo de Donoho y Johnstone se deduce que la elección del umbral u debe optimizarse para cada señal en particular, ya que como puede verse en el segundo paso del algoritmo de filtrado, dicho umbral depende de una constante σ relacionada con el nivel de ruido presente en la señal, valor desconocido en la práctica.

Hemos realizado una implementación del algoritmo en Matlab [Wor96], basándose en un *toolbox* desarrollado en la Universidad de Vigo y denominado UviWave [SGS95]. Dicha implementación tiene por objeto la evaluación del comportamiento de la técnica descrita en el filtrado de señales de voz, al objeto de utilizarla como etapa previa de los algoritmos de extracción de características de la señal que están siendo desarrollados en el Departamento de Informática de la Universidad de Valladolid ([EC98], [SVC98]).

4.2 Implementación del algoritmo propuesto

La implementación del algoritmo de filtrado de señal de voz ha sido desarrollado utilizando como estructura básica el diagrama de bloques que aparece en la figura 4.1.

Como puede verse en dicha figura, la señal obtenida de la grabación (señal en bruto) pasa por una etapa previa de adecuación (1). En esta etapa sólo se efectúa un cambio de formato para su tratamiento con Matlab, no realizándose ninguna modificación en sus componentes frecuenciales ni temporales.

A continuación la señal de trabajo se introduce en el módulo de filtrado (2). El filtrado se realiza sobre la transformada wavelet de la señal, obteniéndose luego, mediante la transformada inversa, una señal libre de ruido. Esta señal se compara con la original en un módulo de evaluación del filtrado (3). Veremos en detalle cada uno de estas tres etapas, analizando en las secciones siguientes los problemas asociados con la elección del criterio de filtrado y del filtro de espejo en cuadratura (QMF) utilizados.

4.2.1 Adecuación de un conjunto de muestras de voz

Se tomaron archivos con muestras de voz del corpus existente en nuestro Departamento, de unos siete segundos de duración y muestreadas a 8 KHz. Esta frecuencia de muestreo suele ser suficiente para capturar toda la información relevante de la señal de voz en condiciones de habla normales.

4.2.2 Desarrollo del módulo de filtrado wavelet

Al realizarse el filtrado en el dominio wavelet, se hace necesario disponer de un conjunto de funciones que calculen la transformada wavelet directa e inversa. Para ello hemos utilizado las funciones desarrolladas por el Departamento de Teoría de Señal de la Universidad de Vigo, y presentes en un toolbox de dominio público denominado UviWave ([SGS95]).

Sea una señal v con 2^J componentes. Para calcular la transformada wavelet de dicha señal se hace necesario disponer de un par de filtros QMF de análisis wavelet h (paso bajo) y g (paso alto) ([RV91]). Con dicho par de filtros puede obtenerse un número k de escalas ($k < J$) de la transformada wavelet de la señal. El vector obtenido tras calcular dicha transformada wavelet -haciendo uso del bloque (2.1) de la figura 4.1- consta de las siguientes partes:

- Los 2^{J-k} primeros componentes del vector son el residuo obtenido tras aplicar el filtro wavelet iterativamente k veces.
- A continuación aparecen los 2^{J-k-1} componentes de la k-ésima escala, los de la escala $k-1$, $k-2$, etcétera, hasta llegar a los 2^{J-1} componentes de la escala 1 (los obtenidos en la primera iteración del algoritmo).

Como puede verse, el módulo WT no separa el resultado del cálculo de cada escala, sino que devuelve un vector con todas las escalas juntas y el residuo correspondiente. Esto ha hecho necesaria la creación de funciones que permitan el tratamiento de la señal en el dominio wavelet.

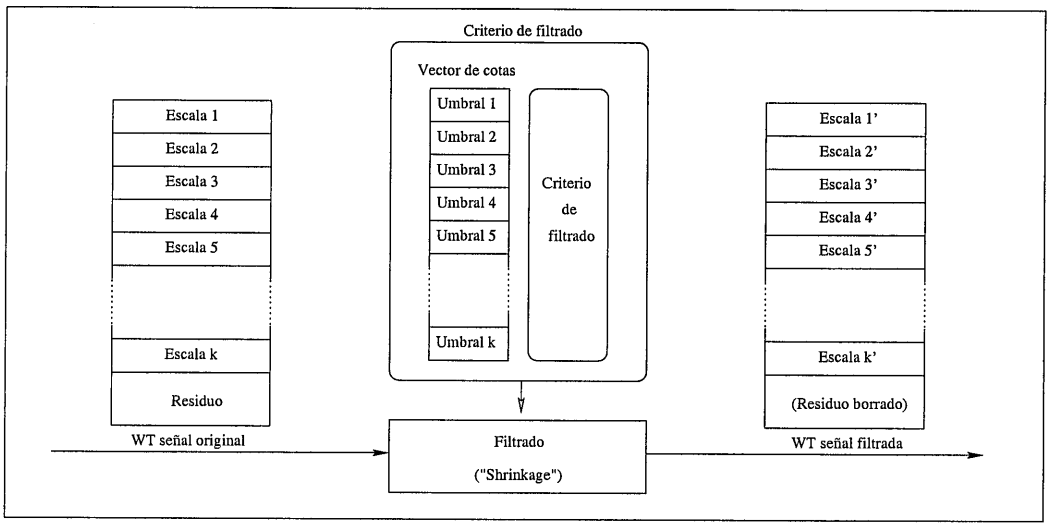

Figura 4.2: Detalle del módulo de filtrado.

La figura 4.2 es una ampliación de la zona enmarcada en línea de puntos de la figura 4.1. El filtrado se realiza a través de un módulo que implementa el criterio *hard thresholding* propuesto por Donoho en [DJ92]. Dicho criterio consiste en tratar cada escala por separado, colocando a cero los coeficientes de dicha escala menores que un umbral determinado y dejando el resto inalterados.

Como puede verse en la figura, se hace necesario disponer de un vector con los umbrales de filtrado para cada escala. En la sección 3 se discuten los posibles criterios de elección de dicho umbral. Una vez hecho esto, el módulo (2.3) de la figura 4.1 calcula la transformada inversa de la señal, obteniéndose nuevamente una representación de la señal de voz en el dominio del tiempo.

4.2.3 Evaluación de la calidad del filtrado

Es evidente que para comparar los resultados obtenidos con diferentes criterios de filtrado es necesario disponer de un método de evaluación que permita determinar la calidad del filtrado realizado. Al igual que se hace en la bibliografía consultada, se utilizó en un principio un criterio estrictamente visual. Para ello se desarrollaron un conjunto de funciones que muestran superpuestas ambas señales, así como su diferencia en valor absoluto y sus correspondientes espectrogramas.

Otro criterio que se ha utilizado en la evaluación del filtrado ha sido comparar la energía de la señal original con la de la señal filtrada. Dado que

el filtrado elimina componentes wavelet antes de realizar la transformación inversa, la energía de la señal reconstruida será menor o igual que la de la señal original. Para la eliminación del ruido en grabaciones lo que se pretende es que la energía de la señal debida al ruido sea cero, mientras que la energía de la señal de voz deberá ser modificada lo menos posible. Por lo tanto, una forma de verificar el filtrado de la señal es calcular la energía de la señal reconstruida, y su diferencia porcentual respecto de la energía de la señal original. Esta diferencia deberá ser despreciable en los segmentos de voz de la señal (ya que la energía de la señal de voz es mucho mayor que la debida al ruido) y máxima en los períodos de silencio de la misma. Este criterio fue el utilizado en el módulo (3) de evaluación del filtrado (figura 4.1).

4.3 Elección del umbral de *thresholding*

Como hemos dicho en la sección anterior, el funcionamiento del proceso de filtrado depende de la elección del umbral de filtrado. Dicho umbral es diferente para cada escala, y en su elección debe tenerse en cuenta las características de las componentes de la señal en esa escala.

Se han ensayado diferentes criterios para la elección del umbral de filtrado. Dichos criterios se expondrán a continuación, examinando el resultado obtenido al aplicar cada uno de ellos. Asimismo, se proponen criterios adicionales, en los que se está trabajando en la actualidad.

4.3.1 Elección del umbral a partir de la energía

El primer criterio utilizado para la elección de los valores de umbral de filtrado en cada escala es el siguiente: elegir los valores sucesivamente para cada escala a partir de la primera de forma que se maximizara el efecto de reducción del ruido en las zonas de silencio de la señal, y se minimizara en las zonas de la señal correspondientes a las vocales (en donde su energía es máxima). Para ello, utilizamos las funciones desarrolladas para la evaluación de la calidad del filtrado, descritas en la sección 4.2.3.

La señal elegida para el ensayo es una señal muestreada a 8 kHz, de unos siete segundos de duración, en la cual una voz de hombre pronuncia la frase en inglés "*seven two six eight zero nine*".

El procedimiento de elección de umbrales fue el siguiente:

1. Para la primera banda se ensayó con diferentes valores de umbral. El valor elegido finalmente fue el valor más pequeño que permitía una reducción máxima del ruido de fondo en las zonas de silencio. Para

valores más altos que él, el ruido en las zonas de silencio no disminuía, y en cambio comenzaba a verse afectada la energía de la señal de voz en más de un cinco por ciento.

2. Una vez establecido el umbral de filtrado para la primera banda, se calculó la transformada de la señal para las dos primeras bandas y, filtrando la señal en la primera con el umbral antedicho, se buscó el umbral de filtrado más adecuado para la segunda, siguendo el mismo criterio.

3. El proceso se repitió hasta la décima escala. A partir de la décima escala los resultados mejoraban eliminando el residuo en lugar de seguir calculando escalas por debajo (en este caso, las escalas 11 y siguiente se corresponde con la banda de frecuencias por debajo de los 8 Hz).

En la figura 4.3 puede observarse el fragmento de la señal correspondiente a la palabra "*six*". Una vez realizado el proceso descrito sobre esta señal, el resultado obtenido fue el siguiente:

- Reducción de la energía de la señal en las zonas de silencio: 99,864 por ciento.
- Reducción de la energía de la señal en las zonas correspondiente a consonantes fricativas: 4,404 por ciento.
- Reducción de la energía de la señal en las zonas de vocales: 0,089 por ciento.

En la figura 4.4 aparece la señal una vez realizado el filtrado. Aunque la mejora visual es evidente, los espectrogramas de ambas señales revelan mejor el comportamiento del filtro (figuras 4.5 y 4.6).

4.3.2 Elección del umbral adaptado a las componentes de la escala

Tras examinar los valores utilizados como umbrales de filtrado de la señal a las diferentes escalas, hemos podido comprobar que en todos los casos el umbral filtraba aproximadamente el 50 por ciento de todos los coeficientes de la escala. Utilizando este criterio de forma estricta -es decir, eligiendo los umbrales de modo que el 50 por ciento de los coeficientes fueran eliminados- los resultados descritos en el apartado anterior mejoraron ligeramente.

El porcentaje de coeficientes a desechar guarda relación con las características de la grabación, ya que si dicha grabación presenta una diferente

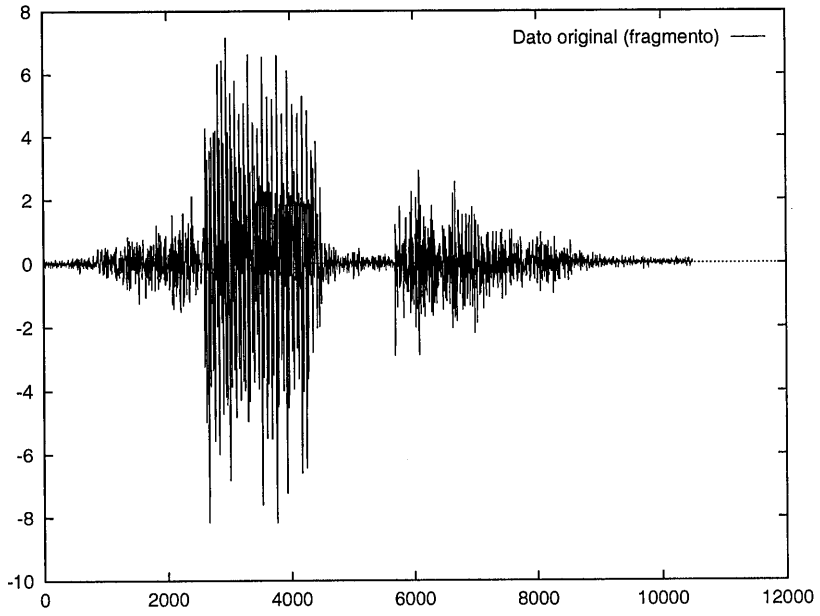

Figura 4.3: Fragmento de la señal correspondiente a "*six*".

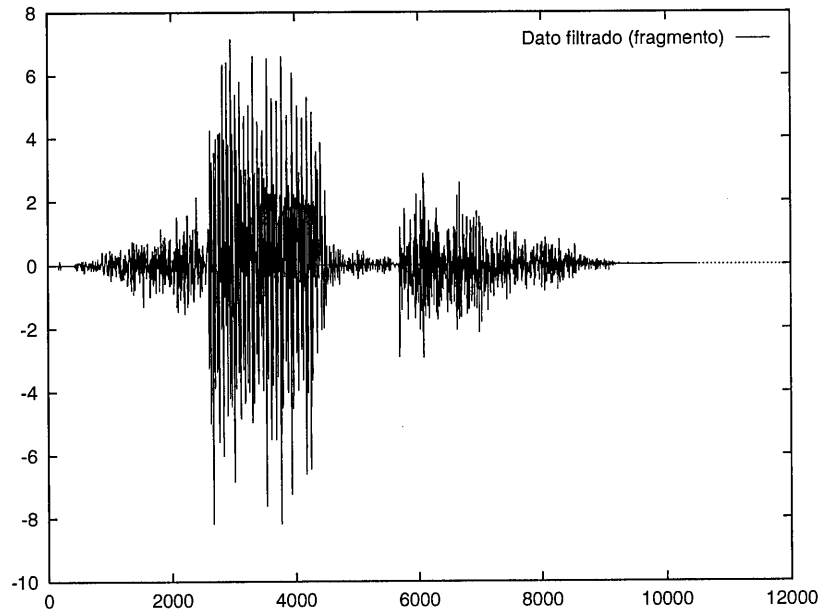

Figura 4.4: Señal de la figura 4.3 una vez filtrada.

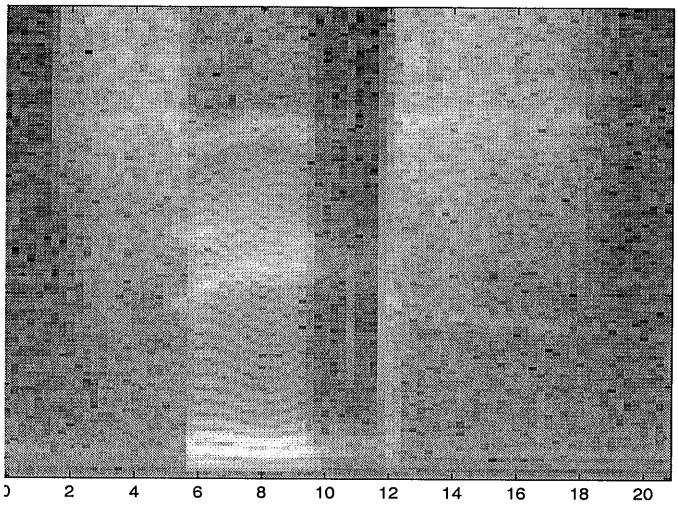

Figura 4.5: Espectrograma de la palabra "*six*" antes del filtrado.

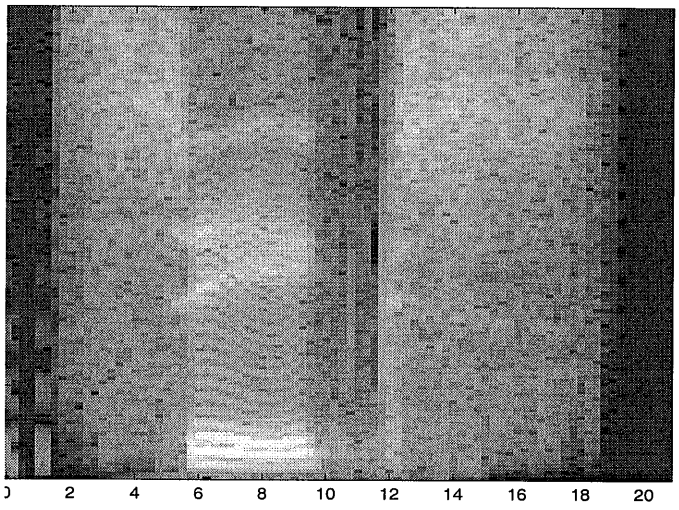

Figura 4.6: Espectrograma de la palabra "*six*" después del filtrado.

proporción de silencios respecto de la duración total, el porcentaje de coeficientes a desechar cambia. Además, dicha proporción también cambia si la grabación presenta un alto nivel de ruido, ya que la eliminación del mismo porcentaje de coeficientes lleva a una pérdida de matices apreciable en la señal.

Por lo tanto, este criterio de eliminar exactamente la mitad de los coeficientes no puede extrapolarse directamente a grabaciones realizadas en otras circunstancias. Sin embargo, a la vista de los resultados parece razonable suponer que para obtener un filtrado adecuado debe eliminarse el mismo porcentaje de coeficientes a todas las escalas analizadas. Estamos trabajando en la cuestión de cómo seleccionar dicho porcentaje.

4.3.3 Utilización del criterio *soft-thresholding*

Se ha implementado también el criterio *soft-thresholding* (propuesto por Donoho [Don92a]). El vector de umbrales utilizado fue el propuesto en la sección anterior. Sin embargo, hemos podido apreciar el problema siguiente: todas las componentes de la señal sufren una atenuación muy fuerte pese a pertenecer a fragmentos de voz. Sin duda, este resultado puede mejorarse utilizando umbrales menores, pero de todos modos la atenuación de todas las componentes persistirá, por la misma naturaleza del criterio de filtrado. Por lo tanto, este criterio parece menos útil en este caso que el de *hard-thresholding*, utilizado en el apartado anterior.

4.3.4 Otros criterios de elección de umbral de filtrado

Los criterios propuestos por Donoho, ya comentados en la sección 1, presentan el inconveniente de que requieren conocer previamente el nivel de ruido de la señal. Dicho valor es desconocido en la práctica, por lo que sólo es posible una estimación utilizando técnicas heurísticas. De obtenerse dicha estimación, podría construirse un filtro adaptativo, que filtrara las componentes de la señal en función de las características de ésta. Este filtro sería de gran utilidad como etapa previa en el procesamiento de señal de voz, por lo que estamos trabajando actualmente en el desarrollo de dicha heurística.

4.4 Elección del filtro QMF a utilizar

Para comparar los efectos del uso de diferentes filtros QMF sobre de la señal, se efectuaron pruebas de filtrado con diferentes filtros propuestos en la bibliografía.

Nombre	Dif. vocales	Dif. fricativas	Dif. silencio	Puntuación
Daubechies 16	0.100124	3.900061	100.000000	20
Daubechies 4	0.092005	4.345299	100.000000	19
Filt3	0.089798	4.404413	99.864239	17
Daubechies 18	0.094582	4.534459	100.000000	13
Symmlet 5	0.094685	3.955151	96.668563	10
Symmlet 4	0.094761	4.453808	100.000000	10
Daubechies 14	0.097866	4.348625	100.000000	10
Coiflet 1	0.095324	4.562462	100.000000	10
Symmlet 7	0.095380	4.379250	99.872295	9
Symmlet 6	0.096476	4.479102	99.870660	8

Tabla 4.1: Comparativa de resultados de filtrado para los diez mejores filtros de entre los 25 analizados. Las diferencias descritas son porcentuales (véase texto).

Se examinaron un total de venticinco filtros: el filtro Haar, el filtro Beylkin, la familia de 5 filtros Coiflet, la familia de 9 filtros Daubechies, la familia de 7 filtros Symmlet y el filtro Vaidyanathan ([Wic94]), además de un filtro biortogonal ("Filt3" de Uviwave) ([SGS95]).

Como criterio de evaluación se ha utilizado el descrito más arriba, esto es, una reducción máxima del ruido y una modificación mínima de la señal hablada. Para este segundo caso se distinguió entre la modificación de la señal correspondiente a vocales y la modificación de la señal correspondiente a sonidos fricativos. La comparativa se realizó como sigue:

- Se realizó el filtrado de una señal utilizando cada uno de los filtros mencionados anteriormente.

- Se utilizó el módulo de evaluación de filtrado para obtener los porcentajes de modificación de la señal filtrada respecto de la señal original, en las tres categorías indicadas (silencio, fricativas y vocales).

- Se ordenaron los filtros según la reducción de la energía en las zonas de silencios en orden decreciente, otorgándose diez puntos al que presentaba una reducción mayor, nueve al siguiente, etcétera.

- Se ordenaron nuevamente los filtros según la reducción de energía en las fricativas, en orden creciente. Se otorgó diez puntos al que menor reducción de la señal presentaba, nueve al siguiente, etcétera.

- Se ordenaron los filtros en función de la reducción de la energía en las vocales, en orden creciente, volviéndose a asignar diez puntos al de menor reducción, y así sucesivamente.

- Finalmente, se sumaron las puntuaciones obtenidas para cada filtro.

Una vez realizado los cálculos con el vector de cotas propuesto en 4.3.2, el resultado final es el que aparece en la tabla 1. En ella puede verse que dos filtros de la familia de Daubechies son los que permiten unos mejores resultados. En concreto, consideramos que el filtro "Daubechies 16" es el mejor de los filtros analizados, al eliminar al completo el ruido modificando la energía de la señal para los sonidos fricativos en menos de un cuatro por ciento.

4.5 Conclusiones

La aplicación de las técnicas vistas al filtrado de señales de voz presenta notables ventajas respecto al filtrado paso bajo convencional, ya que permite eliminar componentes de alta frecuencia de la señal debidos al ruido sin suavizar el resto de detalles de la señal.

Un aspecto importante es el de la elección de los umbrales de filtrado. Como se ha visto, existen diferentes criterios, que podemos clasificar en dos tipos: los que utilizan técnicas heurísticas (por ejemplo, que elimine la energía de los silencios de la grabación), y otros que para su aplicación necesitan conocer el nivel de ruido de la señal, que es precisamente un dato desconocido. Estamos trabajando en la cuestión de la elección automática del umbral de filtrado para cada escala en función de las características de la señal, lo que posibilitaría la obtención de un filtro adaptativo, de gran utilidad como etapa de preproceso de la señal de voz.

La calidad del filtrado realizado depende, como hemos visto, de una correcta elección del filtro QMF que realiza el paso al dominio wavelet de la señal a filtrar. Hemos establecido una comparativa utilizando criterios fácilmente reproducibles, lo que nos ha permitido seleccionar el filtro mejor adaptado a los criterios establecidos de calidad de filtrado.

Entre las posibles aplicaciones de esta técnica figuran su uso como etapa previa a la aplicación sobre la señal de algoritmos de detección de pitch ([EC98]).

Apéndice A

Implementación de funciones Matlab para filtrado de señal de voz

A.1 Introducción

La implementación del algoritmo de filtrado de señal en Matlab comprende los pasos que enumeraremos a continuación.

A.1.1 Obtención de un conjunto de muestras de voz

Se tomaron archivos con muestras de voz del repositorio existente en el Departamento. Estas muestras estaban en ficheros con formato `.116`, por lo que hubo que desarrollar funciones en Matlab que realizaran la conversión. Estas funciones, listadas en la sección A.2, son `mat2l16()` y `l162mat()`.

A.1.2 Desarrollo de funciones de tratamiento de escalas wavelet

Se dispone de un toolbox desarrollado por el Departamento de Teoría de Señal de la Universidad de Vigo denominado UviWave [SGS95]. Dicho toolbox incorpora funciones de cálculo de la transformada Wavelet de una señal dada, así como la definición de algunas transformadas Wavelet básicas. Para un vector v con 2^J componentes, un par de filtros de análisis wavelet h (paso bajo) y g (paso alto) y un número de escalas k (con $k < J$), la función en Matlab

```
wx = wt(x,h,g,k)
```

calcula la transformada wavelet de x para las k primeras escalas y la almacena en el vector wx. El vector así generado consta de las siguientes partes:

- Los 2^{J-k} primeros componentes del vector son el residuo obtenido tras aplicar el filtro wavelet iterativamente k veces.
- A continuación aparecen los 2^{J-k-1} componentes de la k-ésima escala, los de la escala $k-1$, $k-2$, etcétera, hasta llegar a los 2^{J-1} componentes de la escala 1 (los obtenidos en la primera iteración del algoritmo).

Como puede verse, la función `wt` no separa el resultado del cálculo de cada escala, sino que devuelve un vector con todas las escalas juntas y el residuo correspondiente. Esto hizo necesario crear funciones que permitieran el tratamiento de la señal en el dominio Wavelet. Esas funciones son las siguientes:

- Función `extrband(transf, escala)`: Extrae una escala $r \in [1, k]$ de la transformada wavelet de una señal, comprobando además que dicha escala existe (según el número de elementos de los que se componga el vector). Devuelve la escala solicitada.
- Función `filtband(señal, umbral, escala)`: Aplica una contracción "firme" (colocando a cero los coeficientes menores que el umbral y dejando el resto inalterados) de la transformada de una señal que recibe como parámetro, en una escala determinada (también indicada como parámetro). Recibe también el valor umbral de filtrado. Muestra visualmente el resultado.
- Función `vfiltro(señal, cota, qmf)`: Aplica una contracción "firme" (véase comentario función anterior) sobre la transformada Wavelet de una señal recibida como parámetro, sobre N escalas. El número N de escalas a filtrar viene dado por el número de elementos del vector `cota`, que contiene los valores umbrales de filtrado para cada escalas. Puede elegirse un filtro ortogonal a realizar: si no se indica, se utiliza el filtro "filt3" de UviWave. Devuelve la señal filtrada. Muestra además por pantalla el error medio entre la señal original y la filtrada, y la desviación estándar sobre dicho error.
- Función `vfiltror(señal, cota)`: Similar a la función `vfiltro()`, con la diferencia de que pone a cero el residuo tras calcular la transformada Wavelet de la señal para N escalas. Esta operación elimina los componentes de la señal para escalas inferiores a la menor escala filtrada. En

nuestro caso, dichas componentes no son de utilidad en el estudio de la señal.

- Función `vfil000(señal, cota)`: Similar a la función anterior, pero utiliza como criterio de filtrado el de eliminación de un porcentaje de coeficientes en todas las escalas indicado por el usuario.

Evidentemente, para realizar el filtrado de cada escala de la transformada es necesario calcular el valor umbral que va a utilizarse. Volveremos sobre este punto en el apartado 3.

A.1.3 Evaluación del filtrado realizado

Para elegir los umbrales de filtrado para cada escala se hace necesario disponer de un método de análisis que permitiera determinar qué valor de umbral para una escala era el que mejor eliminaba el ruido de la señal (evidente sobre todo en los intervalos de silencio de la grabación), modificando lo menos posible la señal grabada. Para ello se desarrollaron diferentes funciones que permiten comparar la señal original con la señal generada tras el filtrado. Evidentemente, para ello se hace necesario establecer un criterio que permita comparar la calidad del filtrado. Al igual que se hace en la bibliografía consultada, se ha utilizado en primer lugar un criterio estrictamente visual. Las funciones desarrolladas para ello fueron las siguientes:

- Función `evalvisu(señal, fseñal, min, max)`: Utilizada para evaluar de forma visual la calidad del filtrado. Admite como parámetros dos vectores, que representan la señal original y la señal filtrada, y dos valores que indican los valores mínimos y máximos que se desean observar. Muestra en pantalla ambas señales entre dichos valores.

- Función `erroral(señal, fseñal, min, max)`: Esta función recibe los vectores original y reconstruido, y muestra por pantalla la señal original y la diferencia en valores absolutos entre la señal original y la reconstruida, indicando además el error medio. Permite comprobar visualmente la distribución del error, y a qué partes de la señal le afecta en mayor medida.

- Función `mismoal(señal, fseñal, min, max)`: Muestra el vector original y el reconstruido superpuestos, en diferentes colores. Utilizada para evaluar visualmente la calidad del filtrado realizado.

- Función `espectro(dato, rdato)`: Dibuja los espectrogramas correspondientes a las señales `dato` y `rdato`, en tonos de gris.

Otro criterio que se ha utilizado en la evaluación del filtrado ha sido comparar la energía de la señal original con la de la señal filtrada. Dado que el filtrado elimina componentes wavelet antes de realizar la transformación inversa, la energía de la señal reconstruida será menor o igual que la de la señal original. Las funciones desarrolladas para comprobar este extremo son las siguientes:

- Función `iso9000(señal, fseñal, min, max)`: Calcula la energia media entre dos valores de las señales original y reconstruida, y muestra por pantalla el porcentaje de diferencia entre ambas señales. Si este porcentaje es 0, las dos señales son iguales en energía. Si es de 100, la señal reconstruida tiene energia igual a cero. Como comentamos más arriba, el filtrado a través de la técnica de "wavelet shrinkage" nunca aumenta la energía de la señal, ya que elimina la contribución de determinadas componente de escala en la señal reconstruida. Por lo tanto, la energía de la señal reconstruida no puede ser en ningún caso mayor que la energía de la señal original.

- Función `porcenex(señal, fseñal, min, max, ancho, step)`: Genera una gráfica con la señal original en la parte superior, y en la inferior una representacion porcentual de la energia que ha perdido la señal reconstruida. Recibe además como parámetros el radio alrededor del punto en el que se calcula la integral para evaluar la energía y la distancia entre los puntos de cálculo.

A.1.4 Elección del umbral de thresholding

Como hemos dicho en la sección anterior, el funcionamiento del proceso de filtrado depende de la elección del umbral de filtrado. Dicho umbral es diferente para cada escala, y en su elección debe tenerse en cuenta las características de las componentes de la señal en esa escala.

Se han ensayado diferentes criterios para la elección del umbral de filtrado. Dichos criterios se expondrán a continuación, examinando el resultado obtenido al aplicar cada uno de ellos.

A.1.5 Elección del filtro QMF a utilizar

Para comparar los efectos del uso de diferentes filtros QMF sobre de la señal, se efectuaron pruebas de filtrado con diferentes filtros propuestos en la bibliografía.

Para realizar el experimento se escribió una función en Matlab que almacenara los resultados del filtrado en un fichero (función `vfilrout()`) y otra

que la invocara para todos los filtros de espejo en cuadratura disponibles (función `comphard()`).

A.2 Funciones Matlab desarrolladas

A.2.1 Función iso9000()

```
function iso9000 (dato,rdato,c1,c2)

%  function iso9000 (dato,rdato,c1,c2)
%
%  iso9000: Calcula la energia media entre c1 y c2 de las señales
%  'dato' y 'rdato', y muestra por pantalla el porcentaje de diferencia
%  entre ambas señales. Si ese porcentaje es 0, las dos señales poseen
%  la misma energía. Si vale 100, 'rdato' = 0.
%
%  No devuelve nada.
%
%  Diego R. Llanos Ferraris, 18 de Junio de 1997.
%

% Energia señal original:
eneogi = trapz((dato(c1:c2)).*(dato(c1:c2)));

% Energia señal reconstruida:
enerec = trapz((rdato(c1:c2)).*(rdato(c1:c2)));

% Atenuacion de la señal: porcentaje de diferencia en
% las energias original y final.

fprintf('Energía señal original : %f\n',eneogi);
fprintf('Energía señal reconstruida : %f\n',enerec);
dif = eneogi - enerec;
fprintf('Diferencia: %f\n',dif);
porc = (dif * 100) / eneogi;
fprintf('Porcentaje de la diferencia sobre la original: %f\n\n',porc);
```

A.2.2 Función erroral()

```
function erroral (dato, rdato, c1, c2)

%  function erroral (dato, rdato, c1, c2)
%
%  Erroral: muestra una ventana dividida en dos partes: en la
```

```
%  superior la señal 'dato', entre los límites c1 y c2, y en la
%  inferior la diferencia entre la 'dato' y 'rdato', en valor
%  absoluto. No devuelve valores. Usada para comprobar la calidad
%  del filtrado con WT.
%
%  Diego R. Llanos Ferraris, 18 de Junio de 1997.
%

subplot (211), plot (dato(c1:c2)), title ('Señal original');
subplot (212), plot (abs(dato(c1:c2)-rdato(c1:c2)));
errmedio = mean ((dato(c1:c2)-rdato(c1:c2)));
title (['Error en la reconstruccion: ',num2str(errmedio)]);
```

A.2.3 Función evalvisu()

```
function evalvisu (dato, rdato, c1, c2)

%  function evalvisu (dato, rdato, c1, c2)
%
%  Evalvisu : muestra una ventana dividida en dos partes: en la
%  superior la señal 'dato', entre los límites c1 y c2, y en la
%  inferior la señal 'rdato', entre c1 y c2. No devuelve valores.
%  Usada para comprobar la calidad del filtrado con WT.
%
%  Diego R. Llanos Ferraris, 18 de Junio de 1997.
%

subplot (211)
plot (dato(c1:c2))
title (['Señal original entre muestras ',int2str(c1),' y '
   ,int2str(c2)])
subplot (212)
plot (rdato(c1:c2))
title (['Señal reconstruida entre muestras ',int2str(c1),' y '
   ,int2str(c2)])
```

A.2.4 Función extrband()

```
function vector = extrband(transf, escala)

%  function vector = extrband(transf, escala)
%
%  extrband: Extrae la banda escala de la WT transf.
```

```
%
%  Diego R. Llanos Ferraris, 18 de Junio de 1997.
%

tam = length(transf);

% Comprobacion de que la escala solicitada existe.
if ((2^escala) > tam)
fprintf('extrband(): No hay muestras suficientes.\n');
return;
end

% Calculo de los minimos y maximos de la zona de ceros.

if (escala == 1)
min = ceil(tam/2);
max = tam;
vector = transf(min:max);
return;
end

% El resto de los casos:

min = ceil(tam/(2^escala));
vector = transf(min:(2*min));
return;
```

A.2.5 Función filtband()

```
function filtban(senal, cota, banda)

%  function filtban(senal, cota, banda)
%
%  filtban: filtra una señal en la banda indicada, eliminando
%  todos los coeficientes wavelet menores que la cota dada.
%
%  No devuelve nada.
%
%  Requiere UviWave Toolbox (Universidad de Vigo).
%
%  Diego R. Llanos Ferraris, 18 de Junio de 1997.
%

load filt3

% Se calcula la WT.
```

```
fprintf('Calculando la WT para %i banda(s)...\n', banda);
wsenal = wt (senal, h3, g3, banda);

% Se detectan los valores a filtrar.
fprintf('Filtrando la señal...\n');
bf0 = abs(wsenal)>cota;

% Se crea la mascara de filtrado.
[fil col] = size (wsenal);
masc = mascara (col, banda);
filtro = bf0 | masc;

% Se hace el filtrado.
wsenalfilt = wsenal .* filtro;

% Se calcula la IWT.
fprintf('Calculando la IWT para %i banda(s)...\n',banda);
rsenal = iwt (wsenalfilt, rh3, rg3, banda);

% Se calcula el error entre senal y rsenal.
error = abs(senal - rsenal);

% Se calcula el valor máximo del error.
desviacion = std(error);
errormed = mean(error);
fprintf('Error medio = %f\n', errormed);
fprintf('Desviacion estandar = %f\n',desviacion);

% Se muestra las señales y el error
subplot (211), plot(rsenal);
title (['Filtrado banda ',int2str(banda),', cota ',num2str(cota)]);
subplot (212), plot(error);
title (['Distr. del error (medio = ',num2str(errormed),')']);
fprintf('Calculos completos. Examina las señales !\n\n');
```

A.2.6 Función l162mat()

```
function dato = l162mat(nombre)

% function dato = l162mat(nombre)
%
% mat2l16: Convierte un fichero de formato l16 en un vector 'dato'.
%
% dato: la señal convertida.
% nombre: Nombre del fichero donde se lee (en el directorio actual).
%
```

```
% Devuelve el vector obtenido, tras reescalarlo a valores reales
% entre 10 y -10.
%
% Diego R. Llanos Ferraris, 15 de julio de 1997.
%

ampli = 10;

fid = fopen(nombre,'rb');
dato = fread(fid,inf,'integer*2');
dato = rot90(dato);
dato = dato .* (ampli / 32700);
```

A.2.7 Función mat2l16()

```
function vectorl16 = mat2l16(dato, nombre)

% function vectorl16 = mat2l16(dato, nombre)
%
% mat2l16: Convierte un vector 'dato' a formato l16 y lo almacena
% en un fichero de nombre 'nombre'.
%
% dato: la señal a convertir.
% nombre: Nombre del fichero donde se escribe (en el directorio
% actual).
%
% Devuelve el vector l16 obtenido.
%
% Diego R. Llanos Ferraris, 15 de julio de 1997.
%

% ampli = max(abs(dato));
ampli = 10;

vectorl16 = dato .* (32700/ampli);
vectorl16 = round (vectorl16);
fid = fopen(nombre,'wb');
fwrite(fid,vectorl16,'integer*2');
```

A.2.8 Función mismoal()

```
function mismoal (dato, rsenal, c1, c2)
```

```
%  function mismoal (dato, rsenal, c1, c2)
%
%  mismoal: Muestra, superpuestas, las señales 'dato' (en azul) y
%  'rdato' (en amarillo), entre c1 y c2. Utilizada para evaluar la
%  calidad del filtrado.
%
%  No devuelve nada.
%
%  Diego R. Llanos Ferraris, 18 de Junio de 1997.
%

hold on;
plot (dato(c1:c2),'b')
plot (rsenal(c1:c2),'y')
title (['Señales entre ',num2str(c1),' y ',num2str(c2)])
hold off;
```

A.2.9 Función porcene()

```
function porcene (dato,rdato,c1,c2)

%  function porcene (dato,rdato,c1,c2)
%
%  porcene: Pinta en la parte superior la señal 'dato', y en la
%  inferior una representacion porcentual de la energia que ha
%  perdido la señal reconstruida 'rdato'.
%
%  'ancho' indica el intervalo de radio 'ancho' alrededor del
%  cual se calcula la energia.
%  'step' indica cada cuantas muestras se realizara el calculo.
%
%  No devuelve nada.
%
%  Diego R. Llanos Ferraris, 18 de Junio de 1997.
%

ancho = 125;
step  = 100;

muestrario = (c1+ancho+1:step:c2-ancho);

for x = muestrario

eneogi = trapz((dato(x-ancho:x+ancho)).*
(dato(x-ancho:x+ancho)));
```

```
enerec = trapz((rdato(x-ancho:x+ancho)).*
(rdato(x-ancho:x+ancho)));
dif = eneogi - enerec;
porc = (dif * 100) / eneogi;
result = [result porc];
end

subplot(211)
plot (dato(c1:c2)), title ('Señal original')
subplot(212)
plot (result), title ('Diferencia porcentual de energia');
```

A.2.10 Función porcenex()

```
function porcenex (dato,rdato,c1,c2,ancho,step)

%  function porcenex (dato,rdato,c1,c2,ancho,step)
%
%  porcenex: Pinta en la parte superior la señal 'dato', y en la
%  inferior una representacion porcentual de la energia que ha
%  perdido la señal reconstruida 'rdato'. Recibe además el 'ancho'
%  (radio alrededor del punto en el que se calcula la integral),
%  y el 'step' (distancia entre puntos de cálculo).
%
% 'ancho' indica el intervalo de radio 'ancho' alrededor del cual
% se calcula la energia.
% 'step' indica cada cuantas muestras se realizara el calculo.
%
% No devuelve nada.
%
% Diego R. Llanos Ferraris, 18 de Junio de 1997.
%

muestrario = (c1+ancho:step:c2-ancho);

for x = muestrario

eneogi = trapz((dato(x-ancho:x+ancho)).*(dato(x-ancho:x+ancho)));
enerec = trapz((rdato(x-ancho:x+ancho)).*(rdato(x-ancho:x+ancho)));
dif = eneogi - enerec;
porc = (dif * 100) / eneogi;
result = [result porc];
end

subplot(211)
plot (dato(c1:c2)), title ('Señal original')
```

```
subplot(212)
plot (result), title ('Diferencia porcentual de energia');
```

A.2.11 Función vfiltro()

```
function rsenal = vfiltro(senal, cota)

% function rsenal = vfiltro(senal, cota)
%
% vfiltro: Filtrado de un conjunto de bandas, con un filtro
% ortogonal de UviWave.
%
% senal: la señal a filtrar.
% cota: Vector de valores minimos a filtrar.
%
% Devuelve la señal reconstruida.
%
% Requiere UviWave Toolbox (Universidad de Vigo)
%
% Diego R. Llanos Ferraris, 12 de junio de 1997.
%

echo on

load filt3

% Se calcula la WT.
[fil, banda] = size(cota);
fprintf('Calculando la WT para %i banda(s)...\n', banda);
wsenalfilt = wt (senal, h3, g3, banda);


i = 1;
[fil col] = size (wsenalfilt);
for x = cota
% Se detectan los valores a filtrar.
fprintf('Filtrando la banda %i ...\n',i);
bf0 = abs(wsenalfilt)>x;

% Se crea la mascara de filtrado.
masc = mascara (col, i);
filtro = bf0 | masc;

% Se hace el filtrado.
wsenalfilt = wsenalfilt .* filtro;
i = i+1;
```

```
end;

% Se calcula la IWT.
fprintf('Calculando la IWT para %i banda(s)...\n',banda);
rsenal = iwt (wsenalfilt, rh3, rg3, banda);

% Se calcula el error entre senal y rsenal.
error = abs(senal - rsenal);

% Se calcula el valor máximo del error.
desviacion = std(error);
errormed = mean(error);
fprintf('Error medio = %f\n', errormed);
fprintf('Desviacion estandar = %f\n',desviacion);

fprintf('Calculos completos.\n\n');
```

A.2.12 Función vfiltror()

```
function rsenal = vfiltror(senal, cota, qmf)

% function rsenal = vfiltror(senal, cota, qmf)
%
% vfiltror: Filtrado de un conjunto de bandas, con un filtro ortogonal,
% y eliminacion del residuo.
%
% senal: la señal a filtrar.
% cota: Vector de valores minimos a filtrar.
% qmf: Filtro QMF a utilizar (opcional).
%
% Si no se especifica, se usa el filtro FILT3 de UviWave.
%
% Devuelve la señal reconstruida.
%
% Requiere UviWave Toolbox (Universidad de Vigo)
% Diego R. Llanos Ferraris, 12 de junio de 1997.
%

echo on

if nargin<3
load filt3;
else
% Se generan los filtros h3, g3, rh3, rg3.
[h3, g3, rh3, rg3]=QMF2filtro(qmf);
end
```

```
% Se calcula la WT.
[fil, banda] = size(cota);
fprintf('vfiltror: Calculando la WT para %i banda(s)...\n', banda);
wsenalfilt = wt (senal, h3, g3, banda);


i = 1;
[fil col] = size (wsenalfilt);
for x = cota
% Se detectan los valores a filtrar.
fprintf('vfiltror: Filtrando la banda %i ...\n',i);
bf0 = abs(wsenalfilt)>x;

% Se crea la mascara de filtrado.
masc = mascara (col, i);
filtro = bf0 | masc;

% Se hace el filtrado.
wsenalfilt = wsenalfilt .* filtro;
i = i+1;
end;

% Se quita el residuo.
fprintf('vfiltror: Eliminando el residuo...\n');
[fil col] = size(wsenalfilt);
res = (1:1:floor(col/(2^banda)));
for x = res
wsenalfilt(x) = 0;
end;

% Se calcula la IWT.
fprintf('vfiltror: Calculando la IWT para %i banda(s)...\n',banda);
rsenal = iwt (wsenalfilt, rh3, rg3, banda);

% Se calcula el error entre senal y rsenal.
error = abs(senal - rsenal);

% Se calcula el valor máximo del error.
desviacion = std(error);
errormed = mean(error);
fprintf('vfiltror: Error medio = %f\n', errormed);
fprintf('vfiltror: Desviacion estandar = %f\n',desviacion);

% Se muestra las señales y el error
fprintf('vfiltror: Calculos completos. \n');
```

A.2.13 Función vfil000()

```
function rsenal = vfil000(senal,banda,porcentaje)

% function rsenal = vfil000(senal,banda,porcentaje)
%
% vfil000: Filtrado de un conjunto de bandas, con un filtro ortogonal
% de UviWave, y eliminacion del residuo.
%
% Criterio de filtrado: Eliminacion de un porcentaje de los
% coeficientes de cada banda.
%
% senal: la señal a filtrar.
% banda: numero de bandas a filtrar.
% porcentaje: Porcentaje de coeficientes a eliminar.
%
% Devuelve la señal reconstruida.
%
% Requiere UviWave Toolbox (Universidad de Vigo)
%
% Diego R. Llanos Ferraris, 12 de junio de 1997.
%

echo on

load filt3

% Se calcula la WT.
fprintf('vfil000: Calculando la WT para %i banda(s)...\n', banda);
wsenalfilt = wt (senal, h3, g3, banda);


[fil col] = size (wsenalfilt);
cota = (1:banda);
for i = cota
fprintf('vfil000: Calculando corte para banda %i ...\n',i);
% Se aislan los coeficientes de esa banda.
coef = extrband (wsenalfilt,i);

% En coef estan los coeficientes de la banda i.
% Se ordenan los coeficientes y se extrae el valor de corte.
coef=abs(coef);
coef=sort(coef);
corte = coef(length(coef)*(porcentaje/100));

% Se detectan los valores a filtrar.
fprintf('vfil000: Filtrando la banda %i ...\n',i);
bf0 = abs(wsenalfilt)>corte;
```

```
% Se crea la mascara de filtrado.
masc = mascara (col, i);
filtro = bf0 | masc;

% Se hace el filtrado.
wsenalfilt = wsenalfilt .* filtro;
end;

% Se quita el residuo.
fprintf('vfil000: Eliminando el residuo...\n');
[fil col] = size(wsenalfilt);
res = (1:1:floor(col/(2^banda)));
for x = res
wsenalfilt(x) = 0;
end;

% Se calcula la IWT.
fprintf('vfil000: Calculando la IWT para %i banda(s)...\n',banda);
rsenal = iwt (wsenalfilt, rh3, rg3, banda);

% Se calcula el error entre senal y rsenal.
error = abs(senal - rsenal);

% Se calcula el valor máximo del error.
desviacion = std(error);
errormed = mean(error);
fprintf('vfil000: Error medio = %f\n', errormed);
fprintf('vfil000: Desviacion estandar = %f\n',desviacion);

fprintf('vfil000: Calculos completos. \n');
```

A.2.14 Función qmf2filtro()

```
function [h, g, rh, rg]=QMF2filtro(qmf);

% function [h, g, rh, rg]=QMF2filtro(qmf)
%
% QMF2filtro: Convierte un filtro QMF en los cuatro filtros que
% necesita UviWave para analizar y sintetizar la señal.
%
% qmf: filtro QMF.
%
% Diego R. Llanos Ferraris, 28 de enero de 1997.
%
```

```
rh=qmf;

h=fliplr(rh);

rg = -( (-1).^(1:length(rh)) ).*rh;
rg = -1*(fliplr(rg));

g=fliplr(rg);
```

A.2.15 Función vfilsoft()

```
function rsenal = vfilsoft(senal, cota, qmf)

% function rsenal = vfilsoft(senal, cota, qmf)
%
% vfilsoft: Filtrado suave de un conjunto de bandas, con un filtro
% ortogonal, y eliminacion del residuo.
%
% senal: la señal a filtrar.
% cota: Vector de valores minimos a filtrar.
% QMF: Filtro QMF a utilizar (opcional).
%
% Si no se especifica, se usa el filtro FILT3 de UviWave.
%
% Devuelve la señal reconstruida.
%
% Requiere UviWave Toolbox (Universidad de Vigo)
%
% Diego R. Llanos Ferraris, 12 de junio de 1997.
%

echo on

if nargin<3
load filt3;
else
% Se generan los filtros h3, g3, rh3, rg3.
[h3, g3, rh3, rg3]=QMF2filtro(qmf);
end

% Se calcula la WT.
[fil, banda] = size(cota);
fprintf('vfilsoft: Calculando la WT para %i banda(s)...\n', banda);
wsenalfilt = wt (senal, h3, g3, banda);
```

```
i = 1;
[fil col] = size (wsenalfilt);
for x = cota
% Se detectan los valores a filtrar.
fprintf('vfilsoft: Filtrando la banda %i ...\n',i);

bf0 = abs(wsenalfilt)-x;
bf0 = (bf0+abs(bf0))/2;
bf0 = sign(wsenalfilt).*bf0;

% Se enmascara todo menos la banda que interesa.
masc = mascara (col, i);
nuevabanda = bf0.*masc;

% Se pega la banda en la transformada.
wsenalfilt = (wsenalfilt.* (~masc)) + nuevabanda;

i = i+1;
end;

% Se quita el residuo.
fprintf('vfilsoft: Eliminando el residuo...\n');
[fil col] = size(wsenalfilt);
res = (1:1:floor(col/(2^banda)));
for x = res
wsenalfilt(x) = 0;
end;

% Se calcula la IWT.
fprintf('vfilsoft: Calculando la IWT para %i banda(s)...\n',banda);
rsenal = iwt (wsenalfilt, rh3, rg3, banda);

% Se calcula el error entre senal y rsenal.
error = abs(senal - rsenal);

% Se calcula el valor máximo del error.
desviacion = std(error);
errormed = mean(error);
fprintf('vfilsoft: Error medio = %f\n', errormed);
fprintf('vfilsoft: Desviacion estandar = %f\n',desviacion);

% Se muestra las señales y el error
fprintf('vfilsoft: Calculos completos. \n');
```

A.2.16 Función vfilrout()

```
function rsenal = vfilrout(senal, cota, nombre, par)

% function rsenal = vfilrout(senal, cota, nombre, par)
%
% vfilrout: Filtrado de un conjunto de bandas, con un filtro ortogonal,
% y eliminacion del residuo. Igual que vfiltro, pero almacena los
%  datos de salida en un fichero llamado "salida.txt".
%
%  senal: la señal a filtrar.
%  cota: Vector de valores minimos a filtrar.
% nombre: Nombre del filtro QMF a generar con
% makeonfilter (opcional).
% par: Par del filtro a generar (opcional).
%
% Si no se especifica, se usa el filtro FILT3 de UviWave.
%
% Devuelve la señal reconstruida.
%
% Requiere UviWave Toolbox (Universidad de Vigo)
% Diego R. Llanos Ferraris, 12 de junio de 1997.
%

echo on

if nargin<3
load filt3;
nombre = 'filt3';
par = 0;
else
% Se generan los filtros h3, g3, rh3, rg3.
qmf=makeonfilter(nombre,par);
[h3, g3, rh3, rg3]=QMF2filtro(qmf);
end

% Se calcula la WT.
[fil, banda] = size(cota);
fprintf('vfilrout: Filtrando con wavelet %s, par %i...\n',nombre,par);
wsenalfilt = wt (senal, h3, g3, banda);


i = 1;
[fil col] = size (wsenalfilt);
for x = cota
% Se detectan los valores a filtrar.
bf0 = abs(wsenalfilt)>x;

% Se crea la mascara de filtrado.
masc = mascara (col, i);
```

```
filtro = bf0 | masc;

% Se hace el filtrado.
wsenalfilt = wsenalfilt .* filtro;
i = i+1;
end;

% Se quita el residuo.
[fil col] = size(wsenalfilt);
res = (1:1:floor(col/(2^banda)));
for x = res
wsenalfilt(x) = 0;
end;

% Se calcula la IWT.
rsenal = iwt (wsenalfilt, rh3, rg3, banda);

% Se calcula el error entre senal y rsenal.
error = abs(senal - rsenal);

% Se calcula el valor máximo del error.
desviacion = std(error);
errormed = mean(error);
file=fopen('c:\temp\salida.txt','a');
fprintf(file,'\n########################################\n');
fprintf(file,'Vector:\n');
fprintf(file,'%f ',cota);
fprintf(file,
 '\nFiltro usado\tError     Desv.     <<six>>   <<x>>     ruido\n');
fprintf(file,'%s %i\t',nombre,par);
fprintf(file,'%f  ',errormed);
fprintf(file,'%f  ',desviacion);
porc=iso9000s(senal,rsenal,19800,28800);
fprintf(file,'%f  ',porc);
porc=iso9000s(senal,rsenal,27500,28800);
fprintf(file,'%f  ',porc);
porc=iso9000s(senal,rsenal,28800,30000);
fprintf(file,'%f\n',porc);

fclose(file);
```

A.2.17 Función comphard()

```
function comphard(dato,cota)

% function comphard(dato,cota)
```

```
%
% comphard: Compara los resultados de la eleccion de diferentes filtros
% QMF en el filtrado 'hard tresholding'. Recibe el vector 'dato'
% a filtrar, y el vector de cotas de filtrado. Escribe en el
% fichero de salida 'c:/temp/salida.txt'. No devuelve nada.
%
% Requiere UviWave Toolbox (Universidad de Vigo)
%
% Diego R. Llanos Ferraris, 9 de febrero de 1998.
%


vfilrout(dato,cota);
vfilrout(dato,cota,'Haar',2);
vfilrout(dato,cota,'Beylkin',0);
vfilrout(dato,cota,'Coiflet',1);
vfilrout(dato,cota,'Coiflet',2);
vfilrout(dato,cota,'Coiflet',3);
vfilrout(dato,cota,'Coiflet',4);
vfilrout(dato,cota,'Coiflet',5);
vfilrout(dato,cota,'Daubechies',4);
vfilrout(dato,cota,'Daubechies',6);
vfilrout(dato,cota,'Daubechies',8);
vfilrout(dato,cota,'Daubechies',10);
vfilrout(dato,cota,'Daubechies',12);
vfilrout(dato,cota,'Daubechies',14);
vfilrout(dato,cota,'Daubechies',16);
vfilrout(dato,cota,'Daubechies',18);
vfilrout(dato,cota,'Daubechies',20);
vfilrout(dato,cota,'Symmlet',4);
vfilrout(dato,cota,'Symmlet',5);
vfilrout(dato,cota,'Symmlet',6);
vfilrout(dato,cota,'Symmlet',7);
vfilrout(dato,cota,'Symmlet',8);
vfilrout(dato,cota,'Symmlet',9);
vfilrout(dato,cota,'Symmlet',10);
vfilrout(dato,cota,'Vaidyanathan',0);
```

A.2.18 Función espectro()

```
function espectro(dato,rdato);

% function espectro(dato,rdato);
%
% espectro(): Se encarga de dibujar el espectrograma, en tonos de
% gris, de las funciones dato y rdato. Utiliza los valores por
```

```
% defecto de la funcion specgram(), aunque el ancho de la ventana de
% Hamming se ha colocado en 512 en lugar de los 256 por defecto para
% mejorar la resolución. Además, se realiza una pequenya
% correccion para pintar correctamente la gama de grises.
%
% Diego R. Llanos Ferraris, 19 de Febrero de 1998.
%


figure;
[d, fd, td]=specgram(dato,[],512);
imagesc(td,fd,20*log10(abs(0.01+d)));
axis('xy');
colormap gray;

figure;
[r, fr, tr]=specgram(rdato,[],512);
imagesc(tr,fr,20*log10(abs(0.01+r)));
axis('xy');
colormap gray;
```

Bibliografía

[AR77] Jont Allen and Lawrence R. Rabiner. A unified approach to short-time fourier analysis and synthesis. *Proceedings of the IEEE*, 65(11):1558–1564, Noviembre 1977.

[BSRL96] Albert Bijaoui, Eric Slezak, Frederic Rue, and Elena Lega. Wavelets and the study of the distant universe. *Proceedings of the IEEE*, 84(4):670–679, Abril 1996.

[CDJV92] A. Cohen, I. Daubechies, B. Jawerth, and P. Vial. Multiresolution analysis, wavelets, and fast algorithms on an interval. Technical report, Comptes Rendus Acad. Sci. París, 1992.

[Dau88] Ingrid Daubechies. Orthonormal bases of compactly supported wavelets. *Comm. in pure and applied mathematics*, 41(7):909–996, 1988.

[Dau90] Ingrid Daubechies. The wavelet transform, time-frequency localization and signal analysis. *IEEE Transactions on Information Theory*, 36(5):961–1005, Septiembre 1990.

[DJ92] D. L. Donoho and I. M. Johnstone. Ideal space adaptation via wavelet shrinkage. *Tech. Report, Statistics, Stanford*, 1992.

[Don92a] David L. Donoho. De-noising via soft-thresholding. *Tech. Report, Statistics, Stanford*, 1992.

[Don92b] David L. Donoho. Wavelet shrinkage and wavelet-vaguelette decomposition: A 10-minute tour. *International Conference on Wavelets and Applications*, June 1992.

[Don93] David L. Donoho. Nonlinear wavelet methods for recovery of signals, densities, and spectra from indirect and noise data. *Proceedings of Symposium in Applied Mathematics*, 00, 1993.

[EC98] David Escudero and Valentin Cardeñoso. Procesamiento de curvas de frecuencia fundamental. Technical report, Departamento de Informática, Universidad de Valladolid, 1998.

[FKPG96] Marie Farge, Nicholas Kevlahan, Valerie Perrier, and Eric Goirand. Wavelets and turbulence. *Proceedings of the IEEE*, 84(4):639–669, Abril 1996.

[Gab46] D. Gabor. Theory of communication. *Journals of the IEEE*, 93:429–457, 1946.

[Mal96] Stephanie Mallat. Wavelets for a vision. *Proceedings of the IEEE*, 84(4):604–614, Abril 1996.

[Rio91] O. Rioul. Dyadic up-scaling schemes: Simple criteria for regularity. Technical report, SIAM J. Mathematical Analysis, 1991.

[RV91] Oliver Rioul and Martin Vetterli. Wavelets and signal processing. *IEEE SP Magazine*, pages 14–38, Octubre 1991.

[RVH96] Kannan Ramchandran, Martin Vetterli, and Cormac Herley. Wavelets, subband coding and best bases. *Proceedings of the IEEE*, 84(4):541–560, Abril 1996.

[Sch96] Peter Schroder. Wavelets in computer graphics. *Proceedings of the IEEE*, 84(4):615–625, Abril 1996.

[SGS95] S. J. García Galán S. González Sánchez, N. González Prelcic. Uviwave version 3.0 (wavelet toolbox for matlab). *Grupo de Teoría de la Señal, Universidad de Vigo*, 1995.

[SVC98] Hernando Silva-Varela and Valentín Cardeñoso. Phonetic classification in spanish using especialized neural nets hierarchically organized. Technical report, Departamento de Informática, Universidad de Valladolid, 1998.

[VH90] M. Vetterli and C. Herley. Wavelets and filter banks: relationships and new results. In N.M. Alburquerque, editor, *Proc. 1990 Int. Conf. Acoust., Speech, Signal Proc.*, pages 1723–1726, 1990.

[Wic94] Mladen Victor Wickerhauser. *Adapted Wavelet Analysis - From Theory to Software.* IEEE Press, 1 edition, 1994.

[Wor96] The Math Works. *Matlab versión 4 - Guía del Usuario.* Prentice-Hall, 1 edition, 1996.