DEPARTAMENT DE MATEMÀTICA APLICADA

LEARNING MULTIRESOLUTION: TRANSFORMACIONES MULTIESCALA DERIVADAS DE LA TEORÍA DE APRENDIZAJE Y APLICACIONES.

DIONISIO F. YÁÑEZ AVENDAÑO

UNIVERSITAT DE VALÈNCIA
Servei de Publicacions
2011

Aquesta Tesi Doctoral va ser presentada a València el dia 15 de desembre de 2010 davant un tribunal format per:

- Dr. Guillermo Ayala Gallego
- Dr. Albert Cohen
- Dra. Coloma M. Ballester Nicolau
- Dra. Eugenia Martínez Mulada
- Dra. Rosa Donat Beneito

Va ser dirigida per:
Dr. Francesc Aràndiga i Llaudes

Dipòsit legal: V-4127-2011
I.S.B.N.: 978-84-370-8078-9
Edita: Universitat de València
Servei de Publicacions
C/ Arts Gràfiques, 13 baix
46010 València
Spain
Telèfon:(0034)963864115

TESIS DOCTORAL

Learning Multiresolution: Transformaciones multiescala derivadas de la Teoría Estadística de Aprendizaje y aplicaciones

Dionisio F. Yáñez Avendaño

Director: Paco Aràndiga i Llaudes

Universitat de València
Valencia, 2010.

Agradecimientos...

...en mi formación matemática

No podría comenzar estos agradecimientos sin nombrar a mis padres que siempre me han apoyado y siempre me han dicho: "Hijo mío si tú puedes con eso y con mucho más", aunque no sé si lo decían con pleno convencimiento o para animarme.

La primera persona que me hizo disfrutar de las Matemáticas y al que le debo gran parte de mi afición por las ciencias es D. Paco mi profesor de 6º de E. G. B., él me enseñó, me apoyó y me dio consejos que nunca olvidaré. De mi época en el Instituto dar las gracias a mi compañero Alex.

La época estudiando la carrera fue la mejor que jamás he vivido. Aprendí y descubrí la Topología gracias a Rafa Sivera y a Francisca, el Análisis gracias a Mazón y Fuensanta, la Geometría diferencial y el Mathematica gracias a Paco Carreras y el Análisis Numérico gracias a Paco Aràndiga. Y todo acompañado y ayudado por grandes compañeros (y grandes matemáticos), Mario, ¡cuantas tardes haciendo Matemáticas! Rober ¡cuantos consejos sobre Matemáticas!

Aquí comenzó otra etapa, la de doctorado, agradecer a Pep y Rosa todo lo que me han enseñado acerca de la Investigación, cómo escribir, cómo pensar y sobretodo a ser honesto con el trabajo que hago. Gracias a mis compañeras de doctorado: María, M. Carmen y sobretodo a Anna ¡cuántas tostadas saladas hablando de Matemáticas! A los profesores Guillermo Ayala y J. M. Tamarit y a todos los componentes del Dept. de Matemática Aplicada. Muy especialmente dar las gracias a Paco, él me acogió en su familia investigadora. Siempre está ayudándome, enseñándome y guiando mi investigación y mi formación. Muchas gracias Paco.

Por último agradecer a mis compañeras de la U. C. V.: Elena, María, Mª José, Aida y Sonia. Ellas me han hecho reflexionar sobre la enseñanza de las Matemáticas, me han devuelto al mundo real y me han enseñado la importancia de saber transmitir las ideas.

...en lo personal

A todos los que me han ayudado de una forma u otra en el andar rutinario de la vida: alumnos, catecúmenos, compañeros de trabajo, compañeros de congresos, compañeros del fútbol y otros muchos.

Gracias a todos mis compañeros de carrera: Rober, Mario, Elena, Jannet, Rosaura, Pascu, Sahila, Raúl, José Ra, Rebeca, Pablo, Diego, Marta, Rafa, Anabel, Isa, Paula, Víctor. Su ayuda en la carrera fue muy importante para empezar en el mundo de la investigación. Y gracias a los componentes del PAS: Conchín, Amalia y el Jefe (Octavio).

Gracias a todos mis amigos de Albacete que me han aguantado y me han alentado para hacer con alegría esta tesis: José, José el médico, Alex, Simón, Sandra, Fran, Pablo, Yolanda. Sin su paciencia no podría haber acabado este trabajo. Y a mis amigos de Barrax que pese a no aparecer mucho por allí siempre que he ido me han hecho un hueco entre ellos.

Gracias a todos los Dominicos, al Movimiento Juvenil Dominicano: a Socarraets, a Dalit, al Olivar, a la Tropa y al grupo de Sevilla; y a toda la Familia Dominicana. Especialmente gracias a mis amigos Pepe R. OP, Luis OP, Félix OP y Vicente B. OP, sus ánimos han sido determinantes en momentos difíciles.

A los catequistas del Colegio S. Vicente Ferrer PP. Dominicos, la labor allí realizada ha servido en muchos momentos de oxígeno para seguir trabajando. Especialmente gracias a Antonio OP, Paco, Pepe y Santi.

A mi buen amigo Jota con el que he compartido risas, llantos, ánimos y desánimos. Por supuesto, gracias a Elena que ha aguantado todas mis quejas, mis locuras, mis corduras, mis errores y mis defectos y que pese a todo ha continuado a mi lado dándome ánimos y cariño.

No podría haber hecho nada de lo que está aquí sin la ayuda de mi familia, mi Yaya, mis cuñados Javi y Sergio, mis sobrinas Teresa y Martina, mis hermanas Pilar y M^{a} José y sobre todo mis padres Dionisio y Josefina. Todos ellos siempre han estado a mi lado, siempre han sabido sacarme una sonrisa en los momentos complicados, han sabido buscarme cuando estaba perdido, aconsejarme cuando hacía mal las cosas y felicitarme cuando las hacía bien.

Por último, gracias a Él que me dio el consejo más importante de mi vida: "Vivir dándose a los demás", así he conseguido ser feliz.

Valencia, 2010 *Dioni F.*

La Verdad es Verdad
venga de donde venga,
incluso
del mismo diablo.

TOMÁS DE AQUINO OP

Índice general

Introducción general

El tratamiento de señales digitales se ha convertido en los últimos años en una de las tareas más interesantes y de mayor recorrido para la investigación matemática. Hay aplicaciones directas en el campo de la Informática, redes de comunicación, tratamientos médicos, tratamientos de recuperación de obras de arte, de fotografías. Aplicaciones en Física, Mecánica, desarrollos en películas animadas y otras muchas que se conocen y que se conocerán a lo largo del tiempo.

El tratamiento de señales podemos decir que comienza en la época de Fourier (1807), su aplicación en funciones 2π-periódicas y su transformada para señales discretas es utilizada aún hoy con éxito para la compresión y eliminación de ruido ([72, 64]). Sin embargo la transformada de Fourier está deslocalizada en tiempo frecuencia (tan sólo nos ofrece la frecuencia) lo que provocó en los años 80 el desarrollo de las primeras bases *wavelets*. Estas bases tienen una localización tiempo frecuencia y gracias a los filtros que podemos obtener de ellas se pueden utilizar en el tratamiento de señales ([34, 30]).

Los esquemas de subdivisión interpolatorios son reglas que nos permiten refinar un conjunto de datos interpolando los valores intermedios a los puntos dados utilizando combinaciones lineales de los valores vecinos.

Estas dos ideas junto a la resolución de ecuaciones en derivadas parciales es lo que indujo a Harten a elaborar un marco general de multiresolución (ver [51, 52, 53]) que permite por medio de dos operadores fundamentales: decimación, $\mathcal{D}_k^{k-1}$ y predicción, $\mathcal{P}_{k-1}^k$ establecer una conexión entre dos niveles de resolución. La idea de Harten es sencilla pero a su vez está cargada de grandes posibilidades pues generaliza las bases *wavelets* permitiendo la introducción de elementos no lineales en sus operadores.

¿En qué consiste la idea de Harten? En primer lugar, se dio cuenta de que si tenemos un conjunto de valores discretos en un determinado nivel de resolución k, f^k, éstos poseen una naturaleza, es decir, procedían de una cierta función continua f y habían sido discretizados dependiendo de la naturaleza de los datos, así pues generó un operador discretización $\mathcal{D}_k$. Por otra parte si deseamos tener mayor resolución, es decir determinar más puntos, necesitamos reconstruir primero esa señal continua que "*perdimos*" en la decimación por medio de un operador que llamó reconstrucción, $\mathcal{R}_k$ y con estos operadores definió los ya mencionados, así:

$$\mathcal{D}_k^{k-1} = \mathcal{D}_{k-1}\mathcal{R}_k,$$
$$\mathcal{P}_{k-1}^{k} = \mathcal{D}_k\mathcal{R}_{k-1}.$$

Es en el operador $\mathcal{R}_k$ donde se introduce toda la teoría interpolatoria (ver p. ej. [53, 15, 13, 12, 46] y otros muchos) y donde podemos utilizar interpolación no lineal como los métodos presentados en el contexto de solución de ecuaciones diferenciales para capturar las discontinuidades, métodos ENO ver p. ej. [13] y WENO ver p. ej. [17].

Harten impone una serie de condiciones a estos operadores, la primera de ellas es que el operador $\mathcal{D}_k^{k-1}$ sea lineal y sobreyectivo, para ello propone las distintas potencias de la función de Haar $\omega_0(x) = \chi_{[0,1]}$. En la literatura sobre multiresolución podemos encontrar otros operadores decimación no *splines* (ver [46]). Nosotros no trabajaremos en este sentido, fijaremos varios operadores decimación y trabajaremos con ellos. La segunda es que estos operadores cumplan una condición de consistencia: si tenemos una señal f^{k-1} y mejoramos su resolución, es decir, predecimos estos datos $\mathcal{P}_{k-1}^{k} f^{k-1}$ y después decimamos esta predicción entonces recuperaremos los datos iniciales, i. e.

$$\mathcal{D}_k^{k-1}\mathcal{P}_{k-1}^{k} f^{k-1} = f^{k-1}.$$

Sin embargo en algunas aplicaciones (como compresión de imágenes digitales) no necesitamos esta propiedad, en esta memoria se presenta una alternativa para trabajar con operadores no consistentes que ofrece buenos resultados y que conserva las propiedades. Por tanto omitimos esta segunda propiedad que Harten señaló en su marco general.

En esta memoria introducimos otra alternativa al operador reconstrucción. En lugar de utilizar elementos únicamente interpolatorios usamos aproximación por medio de métodos de núcleo. Consisten en aproximar a un cierto valor dependiendo de la cercanía (o lejanía) de los valores de su entorno. Este método generaliza los métodos interpolatorios

introduciendo posibles ventajas al poder utilizar gran cantidad de puntos sin subir el grado del polinomio interpolador. Son muchas las variables que componen un problema de aproximación por métodos de núcleo. En esta memoria estudiamos algunas posibilidades y las ventajas y desventajas que suscitan.

Nos planteamos la siguiente pregunta: conociendo la señal original, ¿por qué no utilizar esta información para generar un operador predictor más adaptativo? Respondemos a ésta utilizando técnicas estadísticas de aprendizaje y generamos predictores que se adaptan a los contornos de la imagen y al nivel de resolución que tenemos. Este tipo de multiresolución nos induce a redefinir algunos conceptos que aparecen en el contexto de multiresolución y que debemos rediseñar para este tipo específico de multiresolución.

Para ambas vías, tanto para multiresolución utilizando métodos de núcleo como para multiresolución de aprendizaje analizamos las distintas propiedades que tienen, las comparamos con los métodos clásicos y mostramos sus resultados.

Esta memoria presenta de manera sencilla dos operadores predicción de multiresolución distintos que abren las puertas a otro gran número de aplicaciones. Durante la realización de estos métodos han surgido diversos problemas. El desarrollo de esta tesis es la solución a dichos problemas.

Planificación de la tesis

Capítulo 1 En el primer capítulo veremos la interpolación segmentaria a trozos fundamental para establecer los métodos clásicos de multiresolución *à la Harten* (ver p. ej. [53, 15]). Además introducimos métodos de interpolación no lineales (ENO y WENO) y presentamos una aplicación para hallar un interpolante monótono siguiendo la interpolación de Hermite. Para esto diseñaremos algunos cambios en ENO y WENO para poder hallar las derivadas de una determinada función. Probaremos algunas de las propiedades que tiene este tipo de interpolación que serán la base para poder probar los distintos problemas que nos surgirán en toda la memoria.

Capítulo 2 En este capítulo comenzamos con la multiresolución *à la Harten*, damos un breve repaso a las distintas variables que conforman el problema y a los métodos clásicos que se pueden utilizar. Presentamos los distintos operadores decimación que usaremos y las propiedades que debe cumplir un operador predicción para que

sea un esquema de multiresolución tal y como lo presentó Harten. Omitiremos una de estas propiedades (la de consistencia) presentando una alternativa bajo una estrategia que llamaremos (AY) y que utilizaremos durante todo el trabajo.

Capítulo 3 Realizamos multiresolución utilizando métodos de núcleo. Para ello definimos en primer lugar dichos métodos, vemos las distintas variables que lo componen y como éstas pueden afectar a nuestros esquemas. Adaptamos este tipo de aproximación a distintos operadores discretización y solucionamos ciertos problemas que nos surgen para esa adaptación. Por último probamos las propiedades que derivan de la multiresolución obteniendo así una gama de métodos nuevos. Generalizamos dichos métodos a dos dimensiones.

Capítulo 4 Introducimos elementos de teoría estadística de aprendizaje a los esquemas de multiresolución creando nuevos métodos que denotamos por LM y que provocan un cambio en la visión de la multiresolución. Adaptamos las definiciones y propiedades propias de la multiresolución a este nuevo tipo y damos las condiciones necesarias para poder diseñar diferentes esquemas. Ofrecemos distintos ejemplos de este método.

1

El problema de la interpolación

Un antiguo teorema de interpolación dice que dos puntos definen una recta, tres puntos una parábola, cuatro una cúbica y así sucesivamente. Los campos de la interpolación y la aproximación han sido cultivados desde hace siglos. En este capítulo mostraremos algunas técnicas de interpolación, poniendo especial interés en técnicas no lineales desarrolladas en los últimos años (ver p.ej. [54, 13]). Son muchos los textos que podemos encontrar sobre el tema de interpolación, aquí citamos algunos [21, 22, 32, 35, 40, 87, 83].

1.1 Introducción

Supongamos que sabemos los valores de una función $f(x)$ en un conjunto de puntos $x_0, \ldots, x_n$, y deseamos calcular el valor de la función en otro punto $\xi \in \overline{\{x_0, \ldots, x_n\}}$; siendo $\overline{\{x_0, \ldots, x_n\}}$ la envoltura convexa de los puntos $x_0, \ldots, x_n$. Los valores $f_i = f(x_i)$ pueden tener distintas interpretaciones, p. ej. medidas físicas o intensidad en el color de un píxel en una imagen. Una forma de conseguir el valor deseado $f(\xi)$ es construir un polinomio interpolador $p_n(x)$ que coincida con los valores de la función en los puntos $x_0, \ldots, x_n$ y evaluar $p_n(\xi)$.

Resulta más sencillo un control efectivo sobre el error global que se comete en un proceso de aproximación por interpolación cuando disponemos de gran cantidad de datos si adoptamos otra estrategia: *interpolación polinómica segmentaria*. La idea básica consiste en dividir el intervalo donde queremos interpolar la función en pequeños subintervalos y utilizar un polinomio diferente (de grado pequeño) en cada uno de estos subintervalos. En este capítulo vamos a definir este tipo de interpolación que se ha utilizado de manera eficiente (ver p. ej. [2, 3, 12, 13, 15, 52, 53]) para diseñar esquemas de multiresolución (capítulo 2) obteniendo en algunos casos filtros similares a los resultantes utilizando técnicas multiescala con base de funciones *wavelets* (ver p. ej. [30, 34, 64, 72]). Entraremos en detalle en algunas características que después utilizaremos en los capítulos 2, 3 y 4 para demostrar los principales resultados de los esquemas de multiresolución que en dichos capítulos describiremos.

Al utilizar interpolación polinómica segmentaria podemos tener un problema al encontrar una discontinuidad[1] en los datos iniciales que producirá en el polinomio interpolador el llamado efecto *gibbs*.

Para solucionar este problema describimos dos técnicas que aparecen en el contexto de soluciones numéricas para Leyes de Conservación hiperbólicas no lineales: *Interpolación esencialmente no oscilatoria* (ENO) introducida por A. Harten et al. (ver p. ej. [51, 54]) que se basa en elegir los puntos interpoladores que eviten (si es posible) la discontinuidad e *Interpolación esencialmente no oscilatoria ponderada* (WENO) que con-

[1] Definimos discontinuidad entre dos puntos x, y cuando el valor absoluto de la diferencia del valor de la función en esos puntos es mayor que una cierta cantidad M que establecemos, i.e.

$$|f(x) - f(y)| > M.$$

siste en utilizar un peso en los distintos interpoladores que se utilizan en ENO. El peso depende de la cercanía de la discontinuidad. Esta técnica fue introducida por Chan et al. en [69] y desarrollada con detalle por Jiang et al. en [65, 17].

Por último en este capítulo explicaremos brevemente la interpolación de Hermite y mostraremos una aplicación utilizando los esquemas ENO y WENO para obtener un operador que conserve la monotonía, ver [9].

1.2 Interpolación polinómica

Las técnicas de interpolación son bien conocidas. En este trabajo tan sólo vamos a rescatar algunas ideas que después utilizaremos en el capítulo 2 para definir un operador en el contexto de la multiresolución. El lector interesado en temas de interpolación y aproximación puede consultar [22, 32, 87, 35] y sus referencias.

Dados $r+1$ puntos, $x_0, \ldots, x_r$ tal que $x_i \neq x_j$ cuando $i \neq j$, existe un único polinomio $q(x)$ de grado r tal que $q(x_j) = f(x_j)$ para $j = 0, \ldots, r$. El conjunto de puntos utilizados para definir $q(x)$ se denomina *stencil* $\mathcal{S}$ de x. Así,

$$f(x) = q(x) + f[x_0, \ldots, x_r, x] \prod_{i=0}^{r} (x - x_j) = f[\mathcal{S}, x] \prod_{x_j \in \mathcal{S}} (x - x_j), \tag{1.1}$$

donde $f[x_0, \ldots, x_r, x] = f[\mathcal{S}, x]$ es la $(r+1)$-ésima diferencia dividida[2] de $f(x)$ en los puntos del *stencil* y x. Si $f(x)$ es suficientemente suave (diferenciable al menos $r+1$ veces) entonces

$$f[x_0, \ldots, x_r, x] = \frac{f^{(r+1)}(\xi_x)}{(r+1)!}, \tag{1.3}$$

donde $\xi_x \in \overline{\{x_0, \ldots, x_r, x\}}$.

[2]Dados $r+1$ puntos, $x_0, \ldots, x_r$, tenemos que:

$$f[x_0, \ldots, x_r] = \frac{f[x_1, \ldots, x_r] - f[x_0, \ldots, x_{r-1}]}{x_r - x_0}, \tag{1.2}$$

con $f[x_i, x_j] = \frac{f(x_i) - f(x_j)}{x_i - x_j}$, $\quad 0 \leq i, j \leq r$.

La diferencia entre $f(x)$ y $q(x)$ nos da información de cómo de "*bueno*" es el interpolante con respecto a la función que es interpolada. Si queremos que esta diferencia sea pequeña sobre un dominio grande una buena estrategia es utilizar interpolación polinómica segmentaria.

Sea $\triangle = \{x_0, \ldots, x_n\}$ un conjunto de puntos igualmente espaciados con $h = x_j - x_{j-1}$, $\forall j = 1, \ldots, n$. Dado un conjunto de datos discretos $f_j = f(x_j)$, la técnica de interpolación polinómica segmentaria consiste en construir un polinomio $q_j(x)$ en cada subintervalo $I_j = [x_{j-1}, x_j]$, $j = 1, \ldots, n$, i.e., seleccionamos un conjuntos de puntos, $\mathcal{S}_j$, para construir $q_j(x)$, para cada j.

El interpolante resultante, $\mathcal{I}(x)$, se obtiene al unir cada trozo, i.e.,

$$\mathcal{I}(x) = q_j(x), \quad x \in [x_{j-1}, x_j],\ j = 1, \ldots, n,$$

cumpliendo siempre la propiedad interpoladora para cada j, $q_j(x_{j-1}) = f_{j-1}$, $q_j(x_j) = f_j$. Esto implica que el punto inicial y el punto final del intervalo I_j debe pertenecer siempre al *stencil* $\mathcal{S}_j$.

En general, si $\mathcal{I}(x)$ es un polinomio interpolante segmentario los polinomios que lo forman en cada trozo son de grado r (i.e., cada *stencil* está compuesto por $r+1$ puntos consecutivos), y si $f(x)$ es suficientemente suave, con la fórmula del error (1.1) junto con (1.3) obtenemos que:

$$\mathcal{I}(x) = f(x) + \mathcal{O}(h^{r+1}), \quad x \in \overline{\{x_0, \ldots, x_n\}}, \tag{1.4}$$

por tanto, las funciones *suaves* pueden ser razonablemente bien aproximadas por técnicas de interpolación segmentaria. El $\mathcal{O}$-término en (1.4) está relacionado con $f^{(r+1)}$, esto nos asegura que las funciones polinómicas de grado menor que r son reconstruidas de manera exacta. Diremos entonces que el *conjunto de exactitud* del esquema interpolatorio incluye todos los polinomios de grado menor que r. Por (1.4) diremos que la técnica de interpolación es de orden $r+1$.

Si $\mathcal{I}(x)$ es el polinomio interpolador a trozos de orden $r+1$ y f es suficientemente suave, es fácil probar que:

$$\frac{d^m}{dx^m}\mathcal{I}(x \pm 0; \mathcal{D}f) = \frac{d^m}{dx^m}f(x) + \mathcal{O}(h^{r+1-m}),\ 0 \leq m \leq r. \tag{1.5}$$

La pérdida de la suavidad en $f(x)$ produce un efecto negativo en el error de interpolación. Consideremos, por ejemplo, que una función $f(x)$ tiene un salto de discontinuidad en el subintervalo j. Es fácil ver que:

$$f[x_{j-1}, x_j] = \frac{f(x_j) - f(x_{j-1})}{x_j - x_{j-1}} = \frac{\mathcal{O}([f])}{h},$$

donde $[f]$ representa el salto de $f(x)$ en I_j. Cualquier diferencia dividida formada por un conjunto de $s+1$ puntos con $s>1$ conteniendo a x_{j-1} y x_j satisface $f[x_l,\ldots,x_{l+s}]=\mathcal{O}([f])/h^s$. Así pues, para cualquier polinomio $q_l(x)$ cuyo *stencil* $\mathcal{S}_l$ contenga a x_{j-1} y x_j, i.e. que cruce la discontinuidad, la fórmula del error (1.1) que obtendremos será

$$f(x)=q_l(x)+\mathcal{O}([f]),\quad x\in I_l,$$

es decir, el interpolador pierde toda exactitud en el intervalo correspondiente.

Si tenemos singularidades en alguna de las derivadas de la función f, la pérdida de exactitud no es tan drástica. Si $f(x)$ tiene un salto de discontinuidad en $f^{(p)}$ en $x_d\in I_j$, cualquier diferencia dividida formada por un conjunto de $s+1$ puntos conteniendo a x_j y x_{j-1} satisface

$$f[x_l,\ldots,x_{l+s}]=\begin{cases}\mathcal{O}([f^{(p)}])/h^{s-p}, & \text{si } s>p;\\ \mathcal{O}(||f^{(s)}||_\infty), & \text{si } s\leq p,\end{cases}\tag{1.6}$$

donde $||f^{(s)}||_\infty$ es la norma máximo de $f^{(s)}$ en la envoltura convexa del *stencil* utilizado para calcular la diferencia dividida, y $[f^{(p)}]=f^{(p)}(x_d^+)-f^{(p)}(x_d^-)$.

Así pues, si $\mathcal{I}(x)$ es un polinomio interpolador segmentario cuyos polinomios a trozos son de grado r, y en algún polinomio $q_l(x)$ hemos utilizado un *stencil* que cruce una discontinuidad tiene el error estimado de la forma

$$f(x)=q_l(x)+\mathcal{O}([f^{(p)}])h^p,\quad \text{cuando}\quad p\leq r.\tag{1.7}$$

Esto nos indica que si tomamos un orden demasiado grande en las zonas suaves tenemos el riesgo de que nuestro *stencil* cruce una discontinuidad y provoque un efecto negativo en la interpolación.

Es obvio que un polinomio no puede aproximar de forma precisa a una función en un intervalo donde ésta contiene una discontinuidad. Así un número relativamente pequeño de discontinuidades aisladas puede degradar la exactitud de un interpolador polinómico segmentario obtenido con una selección del *stencil* lineal e independiente de los datos.

Sin embargo, en los intervalos que no contienen una singularidad la función es suave, por tanto desearíamos utilizar un polinomio a trozos tan preciso como fuese posible. Mientras la envoltura convexa del *stencil* $\overline{\mathcal{S}_j}$ esté en la región de suavidad de la función f, las diferencias divididas son como derivadas (ver (1.3)) y la fórmula del error (1.1) garantiza que obtenemos una precisión exacta. Así, para conseguir esta precisión tenemos que elegir *stencils* que no crucen las discontinuidades, siempre que sea posible.

Harten et al. en [54] introducen una técnica de interpolación segmentaria dependiente de los datos que denomina "*interpolación Esencialmente No Oscilatoria*" (ENO). La intención original con este tipo de interpolación es evitar el fenómeno de *gibbs* que aparece cuando se utiliza interpolación centrada en funciones con discontinuidades de salto. Un interpolante ENO produce un perfil monótono cuando cruza una discontinuidad.

La idea básica en la que se basa la técnica ENO es construir el polinomio a trozos solo con información de las regiones de suavidad de la función interpolada cuando esto sea posible. Si las singularidades están separadas lo suficiente podremos utilizar esta técnica en todos los intervalos, exceptuando, por supuesto, aquellos que contienen las singularidades. Este tipo de interpolación surgido en el contexto de Leyes de Conservación se ha utilizado para construir operadores de multiresolución como veremos en el capítulo 2 (ver [16, 13, 46, 51, 73]).

1.3

Interpolación esencialmente no oscilatoria

La característica esencial de la técnica interpolatoria ENO es el procedimiento selectivo del *stencil*. Para cada subintervalo $I_j = [x_{j-1}, x_j]$, donde f es suave, el objetivo es diseñar una estrategia que elija un *stencil* $\mathcal{S}_j$ que no cruce singularidades. Por tanto necesitamos un conjunto de indicadores de suavidad. Por lo visto en la sección anterior las diferencias divididas (ver (1.4)) pueden utilizarse para este propósito.

Denotamos como

$$\mathcal{S}_j^{ENO} = \{x_{s_j-1}, x_{s_j}, \ldots, x_{s_j+r-1}\}$$

al *stencil* ENO para el j-ésimo intervalo, suponiendo una técnica de orden $r+1$.

En [54] se consideran dos algoritmos para seleccionar el valor s_j:

Algoritmo 1.1. *Selección jerárquica del* stencil

Para cada $j = 1, \ldots, n$
  $s_0 = j$
  for $l = 0, \ldots, r-2$
    $|f[x_{s_l-2}, \ldots, x_{s_l+l}]| < |f[x_{s_l-1}, \ldots, x_{s_l+l+1}]|$ *entonces* $s_{l+1} = s_l - 1$
  end
$s_j = s_{r-1}$

Algoritmo 1.2. *Selección no jerárquica del* stencil

Para cada $j = 1, \ldots, n$ *elegir* s_j *tal que*
$$|f[x_{s_j-1}, \ldots, x_{s_j+r-1}]| = \min\{|f[x_{l-1}, \ldots, x_{l+r-1}]|, j-r+1 \leq l \leq j\}$$

Como en la sección anterior veamos qué sucede cuando la función es suave excepto en una discontinuidad de salto aislada, $x_d \in I_j$. Sea $\mathcal{S}$ un *stencil* de $s+1$ puntos que no cruza la singularidad y $\mathcal{S}^*$ un *stencil* con el mismo número de puntos que la cruce. Por (1.4) tenemos que:

$$f[\mathcal{S}] = \mathcal{O}(1), \qquad f[\mathcal{S}^*] = \mathcal{O}\left(\frac{1}{h^s}\right).$$

Así pues, las diferencias divididas cuyos puntos cruzan la discontinuidad son siempre mayores que las diferencias dividas del mismo orden cuyo *stencil* está contenido en la región de suavidad de la función $f(x)$. Ambos algoritmos eligen los *stencils* que evitan la discontinuidad de salto.

En el caso de tener un "pico" en la función $f(x)$ (es decir una discontinuidad en la derivada de la función, $f'(x)$) la situación es similar. Por (1.4) tenemos que,

$$f[\mathcal{S}] = \mathcal{O}(1), \qquad f[\mathcal{S}^*] = \mathcal{O}\left(\frac{1}{h^{s-1}}\right).$$

Tanto para discontinuidades de salto como para "picos" el procedimiento de selección del *stencil* nos proporciona un polinomio interpolador que verifica:

$$f(x) = q(x) + \mathcal{O}(h^{r+1}), \quad x \in [x_{l-1}, x_l],\ l \leq j-1,\ l \geq j+1,$$

y todos los polinomios en los distintos trozos tienen el orden $r+1$, exceptuando q_j.

La situación es ligeramente diferente para singularidades de orden más alto. El Algoritmo 1.2 determina el *stencil* tan sólo fijándose en las diferencias divididas más grandes. Si la función f es suave en la envoltura convexa del *stencil*, $\overline{\mathcal{S}}$ (con $r+1$ puntos), y $\mathcal{S}^*$ es un *stencil* que cruza la discontinuidad con el mismo número de puntos entonces por (1.4) tenemos que:

$$f[\mathcal{S}] = \mathcal{O}(1), \qquad f[\mathcal{S}^*] = \begin{cases} \mathcal{O}\left(\frac{1}{h^{r-p}}\right), & r > p; \\ \mathcal{O}(1), & r \leq p. \end{cases}$$

Así, para escapar de la discontinuidad de salto en $f^{(p)}$, $p > 1$ necesitamos utilizar este algoritmo no jerárquico (ver discusión del Algoritmo jerárquico 1.1 en [13]) y además necesitamos utilizar un polinomio interpolador de grado r con $r > p$.

Por tanto, si las singularidades de $f(x)$ son discontinuidades de salto o picos y están bien separadas (es posible elegir un *stencil* en la parte suave de la función), entonces la interpolación ENO de grado r (algoritmos 1.1 y 1.2) produce un polinomio interpolador a trozos cumpliendo

$$\frac{d^m}{dx^m}\mathcal{I}(x \pm 0; \mathcal{D}f) = \frac{d^m}{dx^m} f(x) + \mathcal{O}(h^{r+1-m}),\ 0 \leq m \leq r, \tag{1.8}$$

excepto cuando x pertenece a un intervalo conteniendo una singularidad (suficientemente separada). La exactitud del interpolante es máxima en referencia a la región de suavidad.

Para singularidades débiles (es decir, singularidades en las derivadas de la función) la exactitud se mantiene cuando utilizamos el Algoritmo 1.2 y los polinomios de los diferentes intervalos son de grado estrictamente mayor que el orden de la singularidad.

Independientemente del algoritmo que utilicemos un polinomio $q_l(x)$ aproximará a una función $f(x)$ con un orden de precisión bajo en una celda que contiene una singularidad. Cuando la singularidad es de salto, la aproximación en la celda de la malla donde se encuentre será de bajo orden. Para singularidades débiles hay una situación especial donde no

se bajaría el orden: Si la singularidad es un punto de la malla, i.e., $x_d = x_j$. El *stencil* que resultaría si esto ocurre sería:

$$\{x_{j+1}\} \cap \mathcal{S}_l = \emptyset, \quad l \leq j, \qquad \{x_{j-1}\} \cap \mathcal{S}_l = \emptyset, \quad l \geq j+1, \tag{1.9}$$

que garantiza que $q_l(x) = f(x) + \mathcal{O}(h^{r+1})$, $x \in [x_{l-1}, x_l]$, $\forall\, l$, es decir, tenemos exactitud en la aproximación de la función original en $\overline{\{x_0, \ldots, x_n\}}$.

Es improbable que una discontinuidad sea justamente un punto de la malla. Sin embargo, si nosotros dentro de la celda donde se encuentra la discontinuidad sabemos la localización exacta (o una buena aproximación a ella) de la singularidad, la definición del interpolante segmentario $\mathcal{I}(x)$ puede ser modificada para tener la relación $\mathcal{I}(x) = f(x) + \mathcal{O}(h^{r+1})$ válida sobre la region que contiene casi todo el intervalo donde está la singularidad. Ésta es la idea básica en la que se basa Harten en [51] para establecer la técnica de resolución subcelda (SR), que no describimos aquí pero que ha sido utilizada como operador predicción en esquemas de multiresolución en [14, 8, 13, 38, 51].

Una mejora a esta técnica fue introducida por Liu et al. en [69], con el nombre de interpolación esencialmente no oscilatoria ponderada (*Weighted* ENO). La idea consiste básicamente en asignar a cada subintervalo todos los *stencils* de una cierta longitud que lo contengan y construir un polinomio interpolador como combinación convexa de los correspondientes polinomios. Utilizamos la información de todos los nodos contenidos en los *stencils* candidatos en el proceso de selección ENO visto anteriormente con el objetivo de obtener un orden mayor de exactitud en aquellas zonas donde la función sea suave. La clave es elegir de forma razonable los pesos de la combinación convexa. Estos pesos deben ser elegidos dependiendo de si los polinomios atraviesan (o no) una discontinuidad, así pues, tienen (o no) una contribución determinante en el polinomio resultante. Por tanto, los pesos deben variar de acuerdo a la suavidad de la función en los *stencils*, la medida de la suavidad será importante para la definición de estos pesos. Jiang y Shu en [65] presentan una medida de la suavidad más eficiente que la presentada en [69]. En la siguiente sección explicamos esta técnica con detalle. Ambas, tanto ENO como WENO son no lineales; las utilizaremos al final del capítulo para crear un interpolador monótono basado en la interpolación de Hermite (ver [9]). También se han utilizado para construir operadores predicción no lineales en un esquema de multiresolución (ver capítulo 2 y p. ej. [17, 16, 13]). La eficacia de estos métodos en comparación con los obtenidos por medio de interpolación segmentaria a trozos, similares en el contexto de teoría de *wavelets* a las bases biortogonales (BW) obtenidas por Cohen et al. ([30, 34, 72]) la podemos ver en [3, 17, 73].

1.4 Interpolación esencialmente no oscilatoria ponderada

La técnica de interpolación WENO aparece como una mejora de las reconstrucciones ENO de valores puntuales desde medias en celda en el contexto de esquemas de resolución para leyes de conservación. Cuando utilizamos *stencils* de $r+1$ puntos las reconstrucciones ENO, como hemos visto en la sección anterior, dan orden de aproximación $r+1$, excepto en aquellos subintervalos donde se atraviesa una singularidad. Esto es debido al proceso de elección del *stencil* que elige aquél que interpola la función en su zona más suave. Este proceso de selección es sensible al error de redondeo, sin elegir (en algunos casos) el *stencil* más adecuado.

Por otra parte, en lugar de considerar solo el único *stencil* que contiene $r+1$ puntos podríamos utilizar la información de los $2r$ nodos que son necesarios para la selección del *stencil* en el proceso de selección ENO y así obtener una exactitud de orden $2r$ en las regiones en las que la función sea suave. Denotamos $\mathcal{S}^r_{j,s_j}$, $s_j = 1,\dots,r$, los r *stencils* candidatos que contienen x_{j-1} y x_j:

$$\mathcal{S}^r_{j,s_j} = \{x_{j-s_j},\dots,x_{j-s_j+r}\},\ s_j = 1,\dots,r, \tag{1.10}$$

y $q^r_{j,s_j}(x)$ el polinomio interpolador definido utilizando los nodos del *stencil* $\mathcal{S}^r_{j,s_j}$, entonces el interpolante WENO está compuesto por la combinación convexa:

$$q_j(x) = \sum_{s_j=1}^{r} \omega^r_{j,s_j} p^r_{j,s_j}(x) \quad \text{con } \omega^r_{j,s_j} \geq 0,\ s_j = 1,\dots,r, \quad \sum_{s_j=1}^{r} \omega^r_{j,s_j} = 1. \tag{1.11}$$

Se puede utilizar este tipo de interpolación para diseñar un operador predicción no lineal dentro de un esquema de multiresolución (ver capítulo 2, resultados en [17]). Vamos a utilizar los métodos de interpolación ENO como WENO para construir un interpolante monótono. Por tanto, no explicaremos en esta sección algunas proposiciones que nos permiten calcular el interpolante WENO (ver [17]) sino que explicitaremos directamente en la §1.5.5 su utilización en la construcción de dicho interpolante.

1.5

Aplicación de los esquemas ENO y WENO para el cálculo de las derivadas en un interpolante monótono de Hermite

Una aplicación de los esquemas ENO y WENO es la construcción de un interpolante monótono.

En muchos problemas de ingeniería y ciencia (química, robótica, etc.) se demanda que los métodos de aproximación representen la realidad física lo más exactamente posible, por ejemplo si queremos representar una gran cantidad de datos A por una cantidad menor B, conservando algunas propiedades de A. Típicamente, A son datos monótonos y/o convexos y B es un interpolante que toma estos datos y conserva su "forma", es decir, es también monótono y/o convexo. Utilizando métodos *standard*, a veces es necesario sacrificar la interpolación de los datos para conseguir preservar la monotonicidad, o inversamente, sacrificar monotonicidad para preservar interpolación. En muchas de estas aplicaciones además de conservar la monotonía se pide una cierta suavidad en el perfil de la función.

En los últimos años se han desarrollado algunos métodos interpolatorios que preservan la monotonía [45], [63], [44]. En este caso lo que se sacrifica es el orden de interpolación estableciendo cotas superiores para las derivadas. Si los valores aproximados que se obtienen para las derivadas superan cierto valor máximo, éstas se sustituyen por ese valor, de forma que la monotonía del interpolante queda asegurada pero se pierde el control sobre la precisión.

Estudiamos la construcción de un interpolante de tipo Hermite que preserve la monotonía de los datos interpolados sin sacrificar completamente el orden de la interpolación. Para ello utilizamos un procedimiento no lineal para el cálculo de las derivadas y un proceso jerárquico de filtrado, de forma que se obtienen aproximaciones a las derivadas que son del mayor orden posible y producen interpolantes que preservan la monotonía.

1.5.1 Formulación del problema

Sea una partición del intervalo $[x_0, x_n]$, al igual que en la §1.1, $x_0 < \ldots < x_n$ no necesariamente igualmente espaciada. Denotaremos como $\Delta x_j = x_j - x_{j-1}$ al espaciado de la malla y $\Delta f_j = f_j - f_{j-1}$. Las pendientes de las rectas que unen los puntos (x_{j-1}, f_{j-1}) y (x_j, f_j) las denotaremos $m_j = \frac{\Delta f_j}{\Delta x_j}$.

Dados los valores aproximados $\{\dot{f}_j\}$ de la primera derivada de f en los puntos $\{x_j\}$, el interpolante de Hermite **cúbico** se puede escribir como:

$$q_j^3(x) = c_1 + c_2(x - x_{j-1}) + c_3(x - x_{j-1})^2 + c_4(x - x_{j-1})^2(x - x_j), \quad \forall x \in [x_{j-1}, x_j], \tag{1.12}$$

donde:

$$\begin{aligned} c_1 &= f_{j-1}; \\ c_2 &= f[x_{j-1}, x_{j-1}] = \dot{f}_{j-1}; \\ c_3 &= f[x_{j-1}, x_{j-1}, x_j] = \frac{m_j - \dot{f}_j}{\Delta x_j}; \\ c_4 &= f[x_{j-1}, x_{j-1}, x_j, x_j] = \frac{\dot{f}_j + \dot{f}_{j-1} - 2m_j}{(\Delta x_j)^2}. \end{aligned}$$

Por tanto debemos establecer un procedimiento para calcular los valores $\{\dot{f}_j\}$. Para ello utilizaremos técnicas de interpolación no lineal. Veamos antes qué propiedades tienen que cumplir estos valores para que el interpolante sea monótono.

1.5.2 Condiciones para preservar la monotonía

Nos gustaría calcular las derivadas con el mayor orden posible ya que la precisión del interpolante (1.12) será, como máximo, de un orden mayor que él de las derivadas. Lo ideal sería que los valores $\{\dot{f}_j\}$ fuesen de orden h^3 para obtener un interpolante de cuarto orden. Por otra parte queremos que el interpolante conserve las propiedades geométricas de los datos, como la monotonía o la convexidad. Veamos en primer lugar que entendemos por interpolador monótono y qué condiciones deben de cumplir las derivadas.

Definición 1.1. *Sea f una función real y sean $(\alpha, f(\alpha))$, $(\beta, f(\beta))$ dos puntos de su gráfica con $\alpha < \beta$ entonces diremos que un polinomio interpolador, $q(x)$, definido en el intervalo $[\alpha, \beta]$ es monótono*

si $f(\alpha) \leq f(\beta)$ *entonces* $q(\alpha^*) \leq q(\beta^*), \quad \forall\, \alpha^*, \beta^* \in [\alpha, \beta]$ *tal que* $\alpha^* \leq \beta^*$;
si $f(\alpha) \geq f(\beta)$ *entonces* $q(\alpha^*) \geq q(\beta^*), \quad \forall\, \alpha^*, \beta^* \in [\alpha, \beta]$ *tal que* $\alpha^* \leq \beta^*$.

Definición 1.2. *Diremos que un polinomio de Hermite cúbico segmentario definido como*

$$\mathcal{I}^H(x) = q_j(x), \quad x \in [x_{j-1}, x_j], \quad j = 1, \ldots, n; \tag{1.13}$$

con $q_j(x)$ definido en la Ecuación (1.12) es monótono si q_j cumple la Definición 1.1 $\forall\, j$.

Sabiendo el concepto de monotonía del interpolador, veamos las condiciones necesarias y suficientes para que sea monótono.

Teorema 1.1. *Boor-Swartz, [36]*
Si

$$0 \leq \dot{f}_{j-1}, \dot{f}_j \leq 3m_j \quad \text{ó} \quad 3m_j \leq \dot{f}_{j-1}, \dot{f}_j \leq 0, \tag{1.14}$$

entonces el interpolante cúbico de Hermite (1.12) es monótono en $[x_{j-1}, x_j]$.

El criterio puede verse como un cuadrado, llamado "*De Boor- Schwartz region*" vista en la Figura 1.1.

Teorema 1.2. *(Fritsch- Carslon)[45] Condición necesaria*
Sea q_j un interpolante de Hermite monótono de los datos $\{(x_{j-1}, f_{j-1}), (x_j, f_j)\}$, con $\dot{q}_j(x_j) = \dot{f}_j$. Entonces:

$$sign(\dot{f}_{j-1}) = sign(\dot{f}_j) = sign(m_j) \tag{1.15}$$

Además, si $m_j = 0$ entonces q_j es monótono (i.e., constante) si y solo si $\dot{f}_{j-1} = \dot{f}_j = 0$.

Teorema 1.3. *(Fritsch- Carslon)[45] Condición suficiente*
Sea $I_j = [x_{j-1}, x_j]$ y q_j un interpolante cúbico de Hermite de los datos $\{(x_{j-1}, f_{j-1}), (x_j, f_j)\}$, con $\dot{q}_j(x_j) = \dot{f}_j$, y sea,

$$\alpha_j = \frac{\dot{f}_{j-1}}{m_j}, \qquad \beta_j = \frac{\dot{f}_j}{m_j}, \qquad j = 1, ..., n. \tag{1.16}$$

Entonces si:

- *Si $\alpha_j + \beta_j - 2 \leq 0$ entonces q_j es monótono en I_j si y solo (1.15) se cumple.*
- *Si $\alpha_j + \beta_j - 2 \geq 0$ y (1.15) se cumple, entonces q_j es monótono en I_j si y solo si una de las siguientes condiciones se cumple:*

  $2\alpha_j + \beta_j - 3 \leq 0$;
  $\alpha_j + 2\beta_j - 3 \leq 0$;
  $\alpha_j - \frac{1}{3}\frac{(2\alpha_j+\beta_j-3)^2}{\alpha_j+\beta_j-2} \geq 0$.

Se puede ver en la Figura 1.1 que las condiciones del Teorema 1.1 están contenidas en las condiciones del Teorema 1.3. Hay distintos procesos de filtrado como podemos ver en [63], [44] cuyas regiones están contenidas en la "*Fritsch - Carlson monotononicity region*". Utilizaremos dos procesos de filtrado, uno basado en el teorema de Boor-Swartz (restricciones *tipo 1*) y otro basado en el Teorema 1.3 que llamaremos (restricciones *tipo 2*). El coste computacional utilizando estas últimas es mayor pero conseguimos mayor orden en la derivada en algunos de los puntos como veremos en los experimentos numéricos.

1.5.3 Cálculo de las derivadas

Hacemos un breve repaso por las distintas formas que se han utilizado para calcular las derivadas que después utilizaremos para comparar con el nuevo esquema que proponemos basado en las técnicas ENO y WENO.

La primera forma que mostramos es la diseñada por Fritsch y Butland

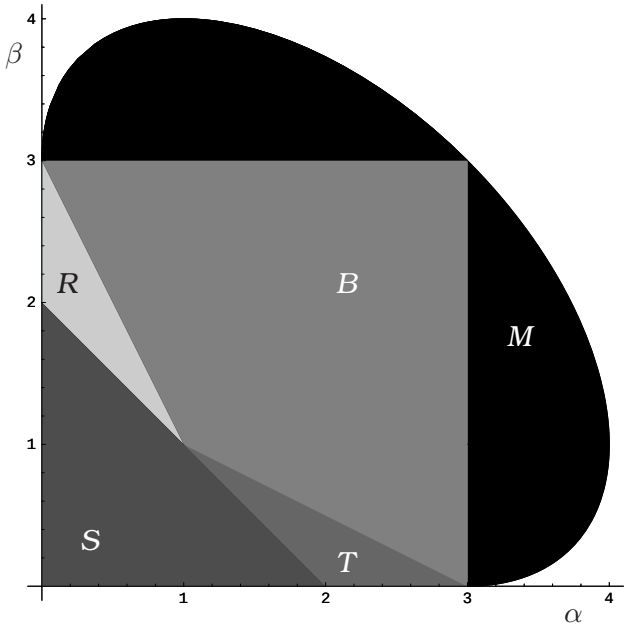


Figura 1.1: *Región B: "De Boor-Swartz box", región S:* $\alpha+\beta-2\leq 0$*, región R:* $\alpha+\beta-2>0$ *y* $2\alpha+\beta-3\leq 0$*, región T:* $\alpha+\beta-2>0$ *y* $\alpha+2\beta-3\leq 0$*, región M:* $\alpha-\frac{1}{3}\frac{(2\alpha+\beta-3)^2}{\alpha+\beta-2}\geq 0$.

en [44]:

$$\dot{f}_j=\begin{cases} sign(m_j)\frac{3|m_j||m_{j+1}|}{|m_j|+2|m_{j+1}|}, & |\ m_{j+1}\ |\leq|\ m_j\ |; \\ sign(m_{j+1})\frac{3|m_j||m_{j+1}|}{|m_{j+1}|+2|m_j|}, & |\ m_{j+1}\ |>|\ m_j\ |; \\ 0, & m_jm_{j+1}\leq 0. \end{cases} \tag{1.17}$$

Esta fórmula cumple las condiciones del Teorema 1.1 y produce aproximaciones de primer orden a los valores de la derivada $\dot{f}(x_j)$, por tanto el interpolante cúbico de Hermite (1.12) es, como máximo, de segundo orden.

Otra alternativa para obtener los valores $\dot{f}_j$, es la aproximación parabólica:

$$\dot{f}_j=\frac{\Delta x_j m_{j+1}+\Delta x_{j+1}m_j}{\Delta x_j+\Delta x_{j+1}}. \tag{1.18}$$

Esta fórmula es de segundo orden. Si evaluamos $\dot{f}_j$, utilizando (1.18) no tenemos garantizado que se preserve la monotonía como veremos en los experimentos numéricos (ver §1.5.7). De hecho, se puede probar que no hay ningún procedimiento lineal (independiente de los datos) capaz

de producir aproximaciones a la derivada de orden mayor que uno de manera que el interpolante resultante conserve la monotonía (ver [36]).

Para asegurar la monotonía del interpolante (1.12) se impone una restricción sobre $\{\dot{f}_j\}$. En primer lugar fijamos el orden de aproximación de las derivadas $\{\dot{f}_j\}$ y proyectamos el resultado sobre el cuadrado de Boor-Swartz (ver [63] y Figura 1.1):

$$\dot{f}_j = \begin{cases} \text{mín}(\text{máx}(0, \dot{f}_j), 3\,\text{mín}(m_j, m_{j+1})), & \text{mín}(m_j, m_{j+1}) > 0; \\ \text{máx}(\text{mín}(0, \dot{f}_j), 3\,\text{máx}(m_j, m_{j+1})), & \text{máx}(m_j, m_{j+1}) < 0; \\ 0, & m_j m_{j+1} \leq 0. \end{cases} \tag{1.19}$$

Una variación de este filtrado es el descrito por Hyman en [63] :

$$\dot{f}_j = \begin{cases} \text{mín}(\text{máx}(0, \dot{f}_j), 3\,\text{mín}(|m_j|, |m_{j+1}|)), & sign(\dot{f}_j) > 0; \\ \text{máx}(\text{mín}(0, \dot{f}_j), -3\,\text{mín}(|m_j|, |m_{j+1}|)), & sign(\dot{f}_j) < 0. \end{cases} \tag{1.20}$$

La diferencia fundamental respecto de (1.19) es que no se impone que la derivada sea cero en las zonas de cambio de monotonicidad, con lo que no se puede asegurar la monotonía en esas zonas a cambio de obtener curvas visualmente más agradables.

1.5.4 Evaluación no lineal de las derivadas I: ENO

Proponemos una fórmula para obtener los valores $\dot{f}_j$ basándonos en la idea de los métodos ENO descritos en la §1.3. El principal cambio que establecemos es que el *stencil* en interpolación debe contener los puntos extremos del intervalo, i.e., si $I_j = [x_{j-1}, x_j]$ entonces $x_{j-1}, x_j \in \mathcal{S}_j^{ENO}$, siendo $\mathcal{S}_j^{ENO}$ uno de los posibles *stencils* que contienen $r+1$ puntos. En en este caso, queremos aproximar la derivada de la función en el punto x_j por tanto seleccionamos un *stencil* con la única condición de que este punto esté contenido en él.

Veamos un ejemplo para clarificar esta pequeña modificación. Sea $r = 3$ entonces tenemos cuatro puntos para interpolar, los posibles *stencils* para el intervalo I_j en el caso interpolatorio (queremos saber el valor del polinomio en el punto $x_{j-\frac{1}{2}}$) son $\mathcal{S}_{j,3}^3 = \{x_{j-3}, x_{j-2}, x_{j-1}, x_j\}$, $\mathcal{S}_{j,2}^3 = \{x_{j-2}, x_{j-1}, x_j, x_{j+1}\}$ y $\mathcal{S}_{j,1}^3 = \{x_{j-1}, x_j, x_{j+1}, x_{j+2}\}$ ya que ningún otro *stencil* formado por cuatro puntos incluye los valor x_{j-1} y x_j. Sin embargo si

queremos calcular la aproximación a la derivada en x_j como no necesitamos que el punto x_{j-1} pertenezca al conjunto de datos tenemos, además de los *stencils* ya mencionados tenemos $\mathcal{S}^3_{j,0} = \{x_j, x_{j+1}, x_{j+2}, x_{j+3}\}$, ver Figura 1.2.

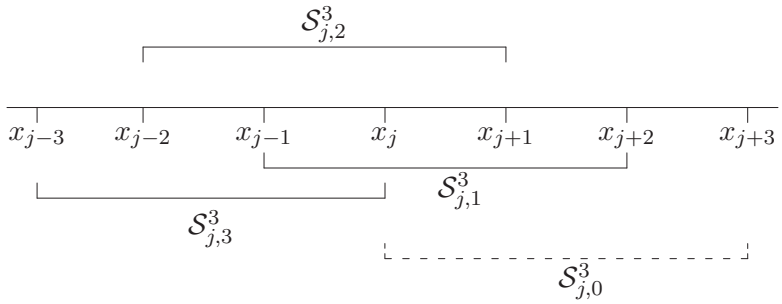


Figura 1.2: Stencils *que se evalúan para la técnica ENO en el* caso interpolatorio*:* $\mathcal{S}^3_{j,0}$*,* $\mathcal{S}^3_{j,1}$ *y* $\mathcal{S}^3_{j,2}$ *y en la aproximación a la derivada* $\mathcal{S}^3_j$*,* $\mathcal{S}^3_{j,1}$*,* $\mathcal{S}^3_{j,2}$ *y* $\mathcal{S}^3_{j,3}$*.*

Así pues, introduciendo este cambio en el Algoritmo jerárquico ENO 1.3 tenemos que:

Algoritmo 1.3. *Algoritmo ENO para obtener el* stencil $\mathcal{S}^r_{j,s_j}$

Para cada $j = 1, \ldots, n$
  $s_0 = 0$
  for $l = 0, \ldots, r-1$
    if $|f[x_{j-s_l-1}, \ldots, x_{j-s_l+l}]| < |f[x_{j-s_l}, \ldots, x_{j-s_l+l+1}]|$
      $s_{l+1} = s_l + 1$
    else
      $s_{l+1} = s_l$
    end
  end
$s_j = s_r$

Por tanto $q_j(x)$ es un polinomio ENO que interpola en los puntos del

stencil $\mathcal{S}^r_{j,s_j} = \{x_{j-s_j}, \ldots, x_{j-s_j+r}\}$ con $0 \leq s_j \leq r$. Como vimos en (1.5) en la §1.3 el valor $\dot{q}_j(x_j)$ es una aproximación de orden r de la derivada de la función f en x_j, siendo f suficientemente suave.

Si $r+1$ es el número de puntos (por tanto la aproximación a la derivada es de orden r) entonces no es difícil calcular las aproximaciones $\dot{f}_{j,s_j}$ para $0 < r \leq 3$ y $0 \leq s_j \leq r$. Así, cuando $r = 1$, tenemos $\dot{f}_{j,1} = m_{j-1}$ y $\dot{f}_{j,0} = m_j$. Si $r = 2$ entonces

$$\begin{aligned} \dot{f}^2_{j,0} &= \tfrac{(2\Delta x_{j+1}+\Delta x_{j+2})m_{j+1}-\Delta x_{j+1}m_{j+2}}{\Delta x_{j+1}+\Delta x_{j+2}}; \\ \dot{f}^2_{j,1} &= \tfrac{\Delta x_j m_{j+1}+\Delta x_{j+1}m_j}{\Delta x_{j+1}+\Delta x_j}; \\ \dot{f}^2_{j,2} &= \tfrac{(2\Delta x_j+\Delta x_{j-1})m_{j+1}-\Delta x_j m_{j-1}}{\Delta x_j+\Delta x_{j-1}} \end{aligned} \tag{1.21}$$

y si $r = 3$ entonces

$$\begin{aligned} \dot{f}^3_{j,0} &= \tfrac{-22f_j+36f_{j+1}-18f_{j+2}+4f_{j+3}}{-22x_j+36x_{j+1}-18x_{j+2}+4x_{j+3}}; \\ \dot{f}^3_{j,1} &= \tfrac{-2f_{j-1}-3f_j+6f_{j+1}-f_{j+2}}{-2f_{j-1}-3f_j+6f_{j+1}-f_{j+2}}; \\ \dot{f}^3_{j,2} &= \tfrac{2f_{j+1}+3f_j-6f_{j-1}+f_{j-2}}{2x_{j+1}+3x_j-6x_{j-1}+x_{j-2}}; \\ \dot{f}^3_{j,3} &= \tfrac{22f_j-36f_{j-1}+18f_{j-2}-4f_{j-3}}{22x_j-36x_{j-1}+18x_{j-2}-4x_{j-3}}. \end{aligned} \tag{1.22}$$

1.5.5 Evaluación no lineal de las derivadas II: WENO

Introducimos al igual que antes para ENO las ideas de los esquemas WENO en el cálculo de las derivadas para obtener un polinomio cúbico de Hermite (1.12) monótono. Al igual que sucedía en la sección anterior, hay una ligera modificación en la elección del *stencil* con respecto a la utilizada habitualmente en la literatura sobre el esquema WENO (ver p. ej. [17], [69]) pero las ideas son las mismas: tomamos $2r + 1$ puntos (no necesariamente igualmente espaciados), por tanto tendríamos el *stencil*

$$\mathcal{S}^{2r}_{j,r} = \{x_{j-r}, \ldots, x_{j+r}\},$$

y sea q^{2r}_j el polinomio que interpola en $\mathcal{S}^{2r}_{j,r}$. Definimos la aproximación a la derivada de la función como

$$\dot{f}_j = \frac{d}{dx} q^{2r}_j(x_j),$$

al igual que antes el orden de aproximación de la derivada en el punto x_j es $2r$ cuando la función f es suficientemente suave. El objetivo de

la técnica WENO es conseguir este orden utilizando una suma convexa de las aproximaciones en los distintos *stencils* que contengan el punto x_j (en el caso de interpolación es el intervalo que contenga a los puntos x_{j-1}, x_j). Así sean los polinomios q^r_{j,s_j} aquellos que interpolan en los *stencils* candidatos

$$\mathcal{S}^r_{j,s_j} = \{x_{j-s_j}, \ldots, x_{j-s_j+r}\},\ s_j = 0, \ldots, r,$$

entonces el interpolador WENO es

$$q^{2r}_j(x) = \sum_{s_j=0}^{r} \omega^r_{j,s_j} q^r_{j,s_j}(x) \quad \text{con } \omega^r_{j,s_j} \geq 0,\ s_j = 0, \ldots, r, \quad \sum_{s_j=0}^{r} \omega^r_{j,s_j} = 1. \quad (1.23)$$

El método WENO para la aproximación de la derivada difiere ligeramente del método WENO para calcular la aproximación en el nodo $\frac{x_{j-1}+x_j}{2}$. Entre otras cosas, los *stencils* no son los mismos.

Ahora debemos calcular los pesos óptimos y los pesos no lineales. Demostraremos las condiciones para que el interpolante llegue al máximo orden de precisión, $2r$, cuando la función sea suave en la región $[x_{j-r}, x_{j+r}]$. También veremos que tenemos orden de aproximación r cuando no es suave en dicha región pero existe al menos un intervalo $[x_{j-s_j}, x_{j-s_j+r}]$, $s_j = 0, \ldots, r$, donde la función si que lo es.

Si la función es suave tenemos que,

$$\frac{d}{dx} q^{2r}_{j,r}(x_j) = \sum_{s_j=0}^{r} C^r_{s_j} \frac{d}{dx} q^r_{j,s_j}(x_j),$$

donde $C^r_{s_j} \geq 0$, $\forall\, s_j$, y $\sum_{s_j=0}^{r} C^r_{s_j} = 1$. Veamos una proposición dando la fórmula de los pesos óptimos.

Proposición 1.1. *Denotando al* stencil *y a los índices como*

$$\mathcal{S}^r_{j,s_j} = \{x_{j-s_j}, \ldots, x_{j-s_j+r}\}, \quad \mathcal{J}^r_{j,s_j} = \{j - s_j, \ldots, j - s_j + r\}$$

entonces los pesos óptimos en el caso de aproximación a la derivada en un conjunto de puntos no igualmente espaciados están dados por la expresión recursiva:

$$\begin{aligned} C^r_r &= \prod_{k=j+1}^{j+r} \frac{x_j - x_k}{x_{j-r} - x_k} = \prod_{k \in \mathcal{J}^{2r}_{j,r} \setminus \mathcal{J}^r_{j,r}} \frac{x_j - x_k}{x_{j-r} - x_k}, \\ C^r_{s_j} &= \prod_{k \in \mathcal{J}^{2r}_{j,r} \setminus \mathcal{J}^r_{j,s_j}} \frac{x_j - x_k}{x_{j-s_j} - x_k} - \\ &\quad - \sum_{i=s_j+1}^{r} C^r_i \prod_{k \in \mathcal{J}^r_{j,i} \setminus \mathcal{J}^r_{j,s_j}} \frac{x_j - x_k}{x_{j-s_j} - x_k} \prod_{k \in \mathcal{J}^r_{j,s_j} \setminus \mathcal{J}^r_{j,i}} \frac{x_{j-s_j} - x_k}{x_j - x_k} \end{aligned} \tag{1.24}$$

con $0 \leq s_j < r$.

Demostración Utilizando la base de Lagrange tenemos que:

$$L^r_{i,s_j}(x) = \prod_{\substack{k=j-s_j \\ k \neq i}}^{j-s_j+r} \frac{x - x_k}{x_i - x_k},$$

y por tanto

$$q^r_{j,s_j}(x) = \sum_{i=j-s_j}^{j-s_j+r} f_i L^r_{i,s_j}(x)$$

así:

$$\frac{d}{dx} q^r_{j,s_j}(x) = \sum_{i=j-s_j}^{j-s_j+r} f_i \frac{d}{dx} L^r_{i,s_j}(x)$$

con

$$\frac{d}{dx} L^r_{i,s_j}(x) = \sum_{s=j-s_j}^{j-s_j+r} \frac{1}{x_i - x_s} \prod_{\substack{k=j-s_j \\ k \neq i,s}}^{j-s_j+r} \frac{x - x_k}{x_i - x_k}.$$

Así:

$$\frac{d}{dx}q^r_{j,s_j}(x_j) = \sum_{\substack{i=j-s_j \\ i\neq j}}^{j-s_j+r}\left(\frac{f_j}{x_j-x_i}+\frac{f_i}{x_i-x_j}\prod_{\substack{k=j-s_j \\ k\neq i,j}}^{j-s_j+r}\frac{x_j-x_k}{x_i-x_k}\right)$$
$$= \sum_{i\in\mathcal{J}^r_{j,s_j}\setminus\{j\}}\left(\frac{f_j}{x_j-x_i}+\frac{f_i}{x_i-x_j}\prod_{k\in\mathcal{J}^r_{j,s_j}\setminus\{i,j\}}\frac{x_j-x_k}{x_i-x_k}\right).$$

Del mismo modo:

$$\frac{d}{dx}q^{2r}_{j,r}(x_j) = \sum_{\substack{i=j-r \\ i\neq j}}^{j+r}\left(\frac{f_j}{x_j-x_i}+\frac{f_i}{x_i-x_j}\prod_{\substack{k=j-r \\ k\neq i,j}}^{j+r}\frac{x_j-x_k}{x_i-x_k}\right)$$
$$= \sum_{i\in\mathcal{J}^{2r}_{j,r}\setminus\{j\}}\left(\frac{f_j}{x_j-x_i}+\frac{f_i}{x_i-x_j}\prod_{k\in\mathcal{J}^{2r}_{j,r}\setminus\{i,j\}}\frac{x_j-x_k}{x_i-x_k}\right).$$

Queremos que nuestros pesos cumplan que

$$\frac{d}{dx}q^{2r}_{j,r}(x_j) = \sum_{s_j=0}^{r} C^r_{s_j}\frac{d}{dx}q^r_{j,s_j}(x_j), \tag{1.25}$$

entonces comprobaremos la fórmula (1.24) utilizando s_j. Si $s_j = r$ entonces el único *stencil* que contiene el punto x_{j-r} es $\mathcal{S}^r_{j,r}$, así igualando el término f_{j-r} en (1.25) tenemos que:

$$\frac{C^r_r}{x_{j-r}-x_j}\prod_{k=j-r+1}^{j-1}\frac{x_j-x_k}{x_{j-r}-x_k} = \frac{1}{x_{j-r}-x_j}\prod_{\substack{k=j-r+1 \\ k\neq j}}^{j+r}\frac{x_j-x_k}{x_{j-r}-x_k},$$

$$C^r_r = \prod_{k=j+1}^{j+r}\frac{x_j-x_k}{x_{j-r}-x_k} = \prod_{k\in\mathcal{J}^{2r}_{j,r}\setminus\mathcal{J}^r_{j,r}}\frac{x_j-x_k}{x_{j-r}-x_k}.$$

Si $s_j = r-1$ entonces los únicos dos *stencils* que contienen al punto x_{j-r+1} son $\mathcal{S}^r_{j,r}$ y $\mathcal{S}^r_{j,r-1}$, así igualando en (1.25) el término f_{j-r+1} tenemos que

$$\frac{C^r_r}{x_{j-r+1}-x_j}\prod_{\substack{k=j-r \\ k\neq j-r+1}}^{j-1}\frac{x_j-x_k}{x_{j-r+1}-x_k}+\frac{C^r_{r-1}}{x_{j-r+1}-x_j}\prod_{\substack{k=j-r+2 \\ k\neq j}}^{j+1}\frac{x_j-x_k}{x_{j-r+1}-x_k} =$$
$$= \frac{1}{x_{j-r+1}-x_j}\prod_{\substack{k=j-r \\ k\neq j,j-r+1}}^{j+r}\frac{x_j-x_k}{x_{j-r+1}-x_k};$$

$$C_{r-1}^r = \prod_{\mathcal{J}_{j,r}^{2r}\setminus\mathcal{J}_{j,r-1}^r} \frac{x_j - x_k}{x_{j-r+1} - x_k} - C_r^r \prod_{k\in\mathcal{J}_{j,r}^r\setminus\mathcal{J}_{j,r-1}^r} \frac{x_j - x_k}{x_{j-r+1} - x_k} \prod_{k\in\mathcal{J}_{j,r-1}^r\setminus\mathcal{J}_{j,r}^r} \frac{x_{j-r+1} - x_k}{x_j - x_k}.$$

Supongamos hallados los valores $C_r^r, C_{r-1}^r \ldots, C_{s_j+1}^r$ con $0 \leq s_j < r$, hallemos $C_{s_j}^r$. El punto x_{j-s_j} está contenido en los *stencils* $\mathcal{S}_{j,r}^r, \mathcal{S}_{j,r-1}^r, \ldots, \mathcal{S}_{j,s_j}^r$, así pues en (1.25) para el término f_{j-s_j} tenemos que

$$\sum_{i=s_j}^{r} \frac{C_i^r}{x_{j-s_j} - x_j} \prod_{k\in\mathcal{J}_{j,i}^r} \frac{x_j - x_k}{x_{j-s_j} - x_k} = \frac{1}{x_{j-k} - x_j} \prod_{k\in\mathcal{J}_{j,r}^{2r}} \frac{x_j - x_k}{x_{j-s_j} - x_k}.$$

Despejando $C_{s_j}^r$:

$$C_{s_j}^r = \prod_{k\in\mathcal{J}_{j,r}^{2r}\setminus\mathcal{J}_{j,s_j}^r} \frac{x_j - x_k}{x_{j-s_j} - x_k} - \sum_{i=s_j}^{r} C_i^r \prod_{k\in\mathcal{J}_{j,i}^r\setminus\mathcal{J}_{j,s_j}^r} \frac{x_j - x_k}{x_{j-s_j} - x_k} \prod_{k\in\mathcal{J}_{j,s_j}^r\setminus\mathcal{J}_{j,i}^r} \frac{x_{j-s_j} - x_k}{x_j - x_k}.$$

■

Los valores de los pesos óptimos para $r = 1, 2$ están recogidos en la Tabla 1.1.

	$r=1$	$r=2$
$\mathcal{S}_{j,r-1}^r$	$\frac{x_{j+1}-x_j}{x_{j+1}-x_{j-1}}$	$\frac{(x_j-x_{j+1})(x_j-x_{j+2})}{(x_{j-2}-x_{j+1})(x_{j-2}-x_{j+2})}$
$\mathcal{S}_{j,r}^r$	$\frac{x_j-x_{j-1}}{x_{j+1}-x_{j-1}}$	$\frac{(x_j-x_{j-2})(x_j-x_{j+2})}{x_{j-2}-x_{j+2}}(\frac{1}{x_{j-1}-x_{j+2}} + \frac{1}{x_{j-2}-x_{j+1}})$
$\mathcal{S}_{j,r+1}^r$		$\frac{(x_j-x_{j-2})(x_j-x_{j-1})}{(x_{j+2}-x_{j-2})(x_{j+2}-x_{j-1})}$

Tabla 1.1: *Pesos óptimos* $r = 1, 2$.

Corolario 1.1. *Los pesos óptimos en el caso de aproximación a la derivada en un conjunto de puntos igualmente espaciados están dados por la expresión:*

$$C_{s_j}^r = \binom{r}{s_j}^2 \binom{2r}{r}^{-1}, \quad 0 \leq s_j \leq r. \tag{1.26}$$

Demostración Hacemos la prueba por inducción sobre s_j. Suponemos sin pérdida de generalidad que $x_0 = 0, \ldots, x_j = jh, \ldots, x_n = nh$. Si $s_j = r$, por la Proposición 1.1 tenemos que:

$$\begin{aligned} C_r^r &= \prod_{k=j+1}^{j+r} \frac{x_j - x_k}{x_{j-r} - x_k} \\ &= \prod_{i=1}^{r} \frac{jh - (j+i)h}{(j-r)h - (j+i)h} \\ &= \prod_{i=1}^{r} \frac{i}{r+i} = \binom{2r}{r}^{-1} = \binom{r}{r}^2 \binom{2r}{r}^{-1}. \end{aligned}$$

Supongamos que se cumple la fórmula (1.26) para $t+1 \leq s_j \leq r$ con $0 \leq t < r$ entonces hallemos C_t^r. Sabemos que:

$$\begin{aligned} \mathcal{J}_{j,r}^{2r} \setminus \mathcal{J}_{j,t}^r &= \{j-r, j-r+1, \ldots, j-t-1, j-t+r+1, \ldots, j+r\}, \\ \mathcal{J}_{j,i}^r \setminus \mathcal{J}_{j,t}^r &= \{j-i, \ldots, j-t-1\}, \\ \mathcal{J}_{j,t}^r \setminus \mathcal{J}_{j,i}^r &= \{j-i+r+1, \ldots, j-t+r\}, \end{aligned} \tag{1.27}$$

$$\begin{aligned} \prod_{k \in \mathcal{J}_{j,r}^{2r} \setminus \mathcal{J}_{j,t}^r} \frac{x_j - x_k}{x_{j-t} - x_k} &= \prod_{s=t+1}^{r} \frac{x_j - x_{j-s}}{x_{j-t} - x_{j-s}} \prod_{s=r+1-t}^{r} \frac{x_j - x_{j+s}}{x_{j-t} - x_{j+s}} \\ &= \prod_{s=t+1}^{r} \frac{s}{s-t} \prod_{s=r+1-t}^{r} \frac{s}{s+t} \\ &= \binom{r}{t} \frac{r!^2}{(r-t)!(r+t)!} = \binom{r+t}{t}^{-1} \binom{r}{t}^2; \end{aligned} \tag{1.28}$$

$$\prod_{k \in \mathcal{J}_{j,i}^r \setminus \mathcal{J}_{j,t}^r} \frac{x_j - x_k}{x_{j-t} - x_k} = \prod_{s=t+1}^{i} \frac{x_j - x_{j-s}}{x_{j-t} - x_{j-s}} = \prod_{s=t+1}^{i} \frac{s}{s-t} = \binom{i}{t}; \tag{1.29}$$

$$\prod_{k \in \mathcal{J}_{j,t}^r \setminus \mathcal{J}_{j,i}^r} \frac{x_{j-t} - x_k}{x_j - x_k} = \prod_{s=r-i+1}^{r-t} \frac{x_{j-t} - x_{j+s}}{x_j - x_{j+s}} = \prod_{s=r-i+1}^{r-t} \frac{s+t}{s} = \frac{r!(r-i)!}{(r-i+t)!(r-t)!}. \tag{1.30}$$

Así como

$$\frac{r!(r-i)!i!}{(r-i+t)!(r-t)!(i-t)!t!} = \binom{r}{t}\binom{r}{i}^{-1}\binom{r}{i-t},$$

$$\sum_{i=t+1}^{r}\binom{r}{i}\binom{r}{i-t} = \frac{(2r)!}{(r-t)!(r+t)!} - \binom{r}{t},$$

por hipótesis de inducción, por (1.28) y por la Prop. 1.1 tenemos que:

$$\begin{aligned}
C_t^r &= \prod_{k\in\mathcal{J}_{j,r}^{2r}\setminus\mathcal{J}_{j,t}^r}\frac{x_j-x_k}{x_{j-t}-x_k} - \sum_{i=t+1}^{r} C_i^r \prod_{k\in\mathcal{J}_{j,i}^r\setminus\mathcal{J}_{j,t}^r}\frac{x_j-x_k}{x_{j-t}-x_k}\prod_{k\in\mathcal{J}_{j,t}^r\setminus\mathcal{J}_{j,i}^r}\frac{x_{j-t}-x_k}{x_j-x_k}\\
&= \binom{r+t}{t}^{-1}\binom{r}{t}^2 - \sum_{i=t+1}^{r}\binom{r}{i}^2\binom{2r}{r}^{-1}\binom{r}{t}\binom{r}{i}^{-1}\binom{r}{i-t}\\
&= \binom{r+t}{t}^{-1}\binom{r}{t}^2 - \binom{r}{t}\binom{2r}{r}^{-1}\sum_{i=t+1}^{r}\binom{r}{i}\binom{r}{i-t}\\
&= \binom{r+t}{t}^{-1}\binom{r}{t}^2 - \frac{(2r)!}{(r-t)!(r+t)!}\binom{r}{t}\binom{2r}{r}^{-1} + \binom{r}{t}^2\binom{2r}{r}^{-1}\\
&= \binom{r+t}{t}^{-1}\binom{r}{t}^2 - \frac{(2r)!}{r!r!}\frac{r!^2}{(r-t)!(r+t)!}\binom{r}{t}\binom{2r}{r}^{-1} + \binom{r}{t}^2\binom{2r}{r}^{-1}\\
&= \binom{r+t}{t}^{-1}\binom{r}{t}^2 - \frac{r!^2}{(r-t)!(r+t)!}\binom{r}{t} + \binom{r}{t}^2\binom{2r}{r}^{-1} = \binom{r}{t}^2\binom{2r}{r}^{-1}.
\end{aligned}$$

■

Explicitamos en la Tabla 1.2 los pesos óptimos para $r = 1, 2$ si los puntos están igualmente espaciados.

	$\mathcal{S}_{j,r}^r$	$\mathcal{S}_{j,r-1}^r$	$\mathcal{S}_{j,r-2}^r$
$r=1$	$\frac{1}{2}$	$\frac{1}{2}$	-
$r=2$	$\frac{1}{6}$	$\frac{2}{3}$	$\frac{1}{6}$

Tabla 1.2: *Pesos óptimos* $r = 1, 2$ *cuando* $\Delta x_j = h$, $\forall\, j$.

Veamos ahora como diseñar ω_{j,s_j}^r para obtener el máximo orden posible. Seguiremos las ideas propuestas en [69, 17, 18] para demostrar la siguiente proposición.

Proposición 1.2. *Sean los pesos no lineales*

$$\omega_{j,s_j}^r = \frac{\alpha_{j,s_j}^r}{\sum_{i=0}^r \alpha_{j,i}^r}, \quad s_j = 0, \ldots, r \quad \textit{donde} \quad \alpha_{j,s_j}^r = \frac{C_{s_j}^r}{(\epsilon + IS_{j,s_j}^r)^2}$$

entonces si IS_{j,s_j}^r es un indicador de suavidad de la función f en el stencil *$\mathcal{S}_{j,s_j}^r$ de orden $\mathcal{O}(h^2)$ cuando la función es suave y cuando la función no es suave no tiende a 0 con h y $\epsilon = \mathcal{O}(h^2)$ entonces se demuestra que si*

$$\omega_{j,s_j}^r = C_{s_j}^r(1 + \mathcal{O}(h^m)) \tag{1.31}$$

con $m \geq r$ entonces:

- *Si f es suave en todos los* stencils *$\mathcal{S}_{j,s_j}^r$, $s_j = 0, \ldots, r$, entonces:*

$$\dot{f}(x_j) - \dot{q}(x_j) = \mathcal{O}(h^{2r}). \tag{1.32}$$

- *Si la función no es suave en todos los* stencils *pero lo es en al menos uno de ellos entonces*

$$\dot{f}(x_j) - \dot{q}(x_j) = \mathcal{O}(h^{\text{mín}(4,r)}). \tag{1.33}$$

Demostración Supongamos en primer lugar que f es suave en todos los *stencils*, así teniendo en cuenta que $r + m \geq 2r$, $\sum_{s_j=0}^r (C_{s_j}^r - \omega_{j,s_j}^r) = 0$, tenemos que

$$\begin{aligned}
\dot{f}(x_j) - \dot{q}(x_j) &= \dot{f}(x_j) - \frac{d}{dx} q_{j,r}^{2r}(x_j) + \frac{d}{dx} q_{j,r}^{2r}(x_j) - \dot{q}(x_j) \\
&= \mathcal{O}(h^{2r}) + \sum_{s_j=0}^r (C_{s_j}^r - \omega_{j,s_j}^r) \frac{d}{dx} q_{j,s_j}^r(x_j) \\
&= \mathcal{O}(h^{2r}) + \sum_{s_j=0}^r (C_{s_j}^r - \omega_{j,s_j}^r) \big(\frac{d}{dx} q_{j,s_j}^r(x_j) - \dot{f}(x_j) \big) \\
&= \mathcal{O}(h^{2r}) + \sum_{s_j=0}^r \mathcal{O}(h^m)\mathcal{O}(h^r) = \mathcal{O}(h^{2r}).
\end{aligned}$$

En segundo lugar, si f tiene una singularidad en al menos un *stencil* $\mathcal{S}_{j,s_j}^r$, $s_j = 0, \ldots, r$, y además $IS_{j,s_j}^r \nrightarrow 0$ en aquellos *stencils* donde existe la

discontinuidad e $IS^r_{j,s_j} = \mathcal{O}(h^2)$ en otro caso, entonces:

$$\alpha^r_{j,s_j} = \begin{cases} \mathcal{O}(1), & \text{si } f \text{ no es suave en } \mathcal{S}^r_{j,s_j}; \\ \mathcal{O}(h^{-4}), & \text{si } f \text{ es suave en } \mathcal{S}^r_{j,s_j}. \end{cases} \tag{1.34}$$

entonces $\sum_{s_j=0}^r \alpha^r_{j,s_j} = \mathcal{O}(h^{-4})$, y $\omega^r_{j,s_j} = \mathcal{O}(h^4)$ si $\mathcal{S}_{j,s_j}$ no es suave. Sea $\mathcal{K} = \{s_j | \, f \text{ no es suave en } \mathcal{S}_{j,s_j}\}$, entonces

$$\begin{aligned} \dot{f}(x_j) - \dot{q}(x_j) &= \sum_{s_j=0}^{r} \omega^r_{j,s_j} \big(\dot{f}(x_j) - \frac{d}{dx} q^r_{j,s_j}(x_j)\big) \\ &= \sum_{s_j \in \mathcal{K}} \mathcal{O}(h^4)\mathcal{O}(1) + \sum_{s_j \notin \mathcal{K}} \mathcal{O}(1)\mathcal{O}(h^r) = \mathcal{O}(h^{\text{mín}(4,r)}) \end{aligned}$$

■

Nota 1.1. *Si $\epsilon = \mathcal{O}(h^2)$ y $r \leq 2$, entonces la Ec. (1.33) implica que el orden de aproximación del interpolante WENO a la derivada de la función en el punto x_j es r, al menos tan bueno como el interpolante ENO.*

Indicadores de suavidad

En [65] Jiang y Shu definen un indicador diferente al propuesto por Liu et al. en [69] de la siguiente forma:

Si denotamos como:

$$x_{j-1/2} = \frac{x_{j-1} + x_j}{2}, \quad x_{j+1/2} = \frac{x_j + x_{j+1}}{2},$$

entonces el indicador de suavidad de Jiang y Shu adaptado a nuestro caso es:

$$IS^r_{j,s_j} = \sum_{s=1}^{r} \int_{x_{j-1/2}}^{x_{j+1/2}} (x_{j+1/2} - x_{j-1/2})^{2s-1} (q^{(s)}_{j,s_j}(x))^2 dx. \tag{1.35}$$

Esta medida de la suavidad se basa en la medida de la variación total, pero tomando la norma $\mathcal{L}^2$ en lugar de la norma $\mathcal{L}^1$.

Si desarrollamos por Taylor (1.35), obtenemos (si $\dot{f}(x_{j-\frac{1}{2}}) \neq 0$)

$$IS^r_{j,s_j} = \big(h\dot{f}(x_{j-\frac{1}{2}})\big)^2 \big(1 + \mathcal{O}(h^2)\big), \quad s_j = 0, \ldots, r-1 \tag{1.36}$$

Por tanto tenemos asegurado el máxima precisión para $r = 2, 3$, (orden óptimo 3 para $r = 2$ y 5 para $r = 3$).

Indicadores de suavidad de Jiang y Shu para $r = 1, 2$

Consideramos los polinomios interpoladores utilizando los *stencils* $S^1_{j,1} = \{x_{j-1}, x_j\}, S^1_{j,0} = \{x_j, x_{j+1}\}$, así:

$$q^1_{j,1}(x) = f_j + \frac{f_j - f_{j-1}}{x_j - x_{j-1}}(x - x_j) \text{ y } q^1_{j,0}(x) = f_j + \frac{f_{j+1} - f_j}{x_{j+1} - x_j}(x - x_j).$$

Entonces:

$$\dot{q}^1_{j,1}(x) = \frac{f_j - f_{j-1}}{x_j - x_{j-1}} \text{ y } \dot{q}^1_{j,0}(x) = \frac{f_{j+1} - f_j}{x_{j+1} - x_j}.$$

Así:

$$IS^1_{j,s_j} = (x_{j+1/2} - x_{j-1/2})^2 \left(\frac{f_{j-s_j+1} - f_{j-s_j}}{x_{j-s_j+1} - x_{j-s_j}} \right)^2, \quad s_j = 0, 1.$$

Cuando los nodos son igualmente espaciados obtenemos los indicadores de Jiang y Shu ([65]):

$$IS^1_{j,s_j} = (f_{j-s_j+1} - f_{j-s_j})^2, \quad s_j = 0, 1.$$

Consideramos los polinomios interpoladores utilizando los *stencils* $S^2_{j,2} = \{x_{j-2}, x_{j-1}, x_j\}, S^2_{j,1} = \{x_{j-1}, x_j, x_{j+1}\}$ y $S^2_{j,0} = \{x_j, x_{j+1}, x_{j+2}\}$. Sea el polinomio que interpola:

$$q^2_{j,s_j}(x) = a_{s_j} + \frac{b_{s_j}(x - x_j)}{2(x_{j+1/2} - x_{j-1/2})} + \frac{c_{s_j}}{2} \left(\frac{x - x_j}{x_{j+1/2} - x_{j-1/2}} \right)^2, \tag{1.37}$$

entonces:

$$\begin{aligned} (q^2_{j,s_j})'(x) &= \frac{b_{s_j}}{2(x_{j+1/2} - x_{j-1/2})} + \frac{c_{s_j}(x - x_j)}{(x_{j+1/2} - x_{j-1/2})^2} \\ (q^2_{j,s_j})''(x) &= \frac{c_{s_j}}{(x_{j+1/2} - x_{j-1/2})^2}, \end{aligned}$$

donde:

$$a_2 = a_1 = a_0 = f_j,$$

$$
\begin{aligned}
b_2 &= \frac{2(x_{j+1/2}-x_{j-1/2})(x_j-x_{j-1})}{(x_{j-1}-x_{j-2})(x_j-x_{j-2})}f_{j-2} - \frac{2(x_j-x_{j-2})(x_{j+1/2}-x_{j-1/2})}{(x_{j-1}-x_{j-2})(x_j-x_{j-1})}f_{j-1} \\
&\quad + \frac{2(2x_j-x_{j-1}-x_{j-2})(x_{j+1/2}-x_{j-1/2})}{(x_j-x_{j-2})(x_j-x_{j-1})}f_j, \\
b_1 &= \frac{2(x_{j+1/2}-x_{1+1/2})(x_j-x_{j+1})}{(x_{j+1}-x_{j-1})(x_j-x_{j-1})}f_{j-1} + \frac{2(x_{j+1/2}-x_{j-1/2})(x_{j-1}-2x_j+x_{j+1})}{(x_j-x_{j-1})(x_{j+1}-x_j)}f_j + \\
&\quad + \frac{2(x_{j+1/2}-x_{j-1/2})(x_j-x_{j-1})}{(x_{j+1}-x_{j-1})(x_{j+1}-x_j)}f_{j+1}, \\
b_0 &= \frac{2(-x_{j+1}+2x_j-x_{j+2})(x_{j+1/2}-x_{j-1/2})}{(x_{j+2}-x_j)(x_{j+1}-x_j)}f_j + \frac{2(x_{j+2}-x_j)(x_{j+1/2}-x_{j-1/2})}{(x_{j+2}-x_{j+1})(x_{j+1}-x_j)}f_{j+1} \\
&\quad - \frac{2(x_{j+1/2}-x_{j-1/2})(x_{j+1}-x_j)}{(x_{j+2}-x_{j+1})(x_{j+2}-x_j)}f_{j+2}
\end{aligned}
$$

y

$$
\begin{aligned}
c_2 &= \frac{2(x_{j+1/2}-x_{j-1/2})^2}{(x_{j-1}-x_{j-2})(x_j-x_{j-2})}f_{j-2} - \frac{2(x_{j+1/2}-x_{j-1/2})^2}{(x_{j-1}-x_{j-2})(x_j-x_{j-1})}f_{j-1} \\
&\quad + \frac{2(x_{j+1/2}-x_{j-1/2})^2}{(x_j-x_{j-2})(x_j-x_{j-1})}f_j, \\
c_1 &= \frac{2(x_{j+1/2}-x_{j-1/2})^2}{(x_{j+1}-x_{j-1})(x_j-x_{j-1})}f_{j-1} - \frac{2(x_{j+1/2}-x_{j-1/2})^2}{(x_j-x_{j-1})(x_{j+1}-x_j)}f_j \\
&\quad + \frac{2(x_{j+1/2}-x_{j-1/2})^2}{(x_{j+1}-x_{j-1})(x_{j+1}-x_j)}f_{j+1}, \\
c_0 &= \frac{2(x_{j+1/2}-x_{j-1/2})^2}{(x_{j+1}-x_{j+2})(x_j-x_{j+2})}f_{j+2} - \frac{2(x_{j+1/2}-x_{j-1/2})^2}{(x_{j+2}-x_{j+1})(x_{j+1}-x_j)}f_{j+1} \\
&\quad + \frac{2(x_{j+1/2}-x_{j-1/2})^2}{(x_{j+2}-x_j)(x_{j+1}-x_j)}f_j.
\end{aligned}
$$

Entonces tenemos que:

$$\begin{aligned} IS^2_{j,s_j} &= \int_{x_{j-1/2}}^{x_{j+1/2}} (x_{j+1/2}-x_{j-1/2})\left(\frac{b_{s_j}}{2(x_{j+1/2}-x_{j-1/2})}+\frac{c_{s_j}(x-x_j)}{(x_{j+1/2}-x_{j-1/2})^2}\right)^2 + \\ &\quad + \int_{x_{j-1/2}}^{x_{j+1/2}} (x_{j+1/2}-x_{j-1/2})^3 \frac{c_{s_j}^2}{(x_{j+1/2}-x_{j-1/2})^4} \\ &= c_{s_j}^2 + \frac{b^2}{4} + \frac{b_{s_j}c_{s_j}}{2}\frac{(x_{j+1/2}-2x_j+x_{j-1/2})}{(x_{j+1/2}-x_{j-1/2})} \\ &\quad + \frac{c_{s_j}^2}{3}\left(\frac{(x_{j+1/2}-x_j)^3-(x_{j-1/2}-x_j)^3}{(x_{j+1/2}-x_{j-1/2})^3}\right). \end{aligned}$$

Observamos que si los nodos están igualmente espaciados tenemos:

$$IS^2_{j,2} = \frac{1}{4}(f_{j-2}-4f_{j-1}+3f_j)^2+\frac{13}{12}(f_{j-2}-2f_{j-1}+f_j)^2, \quad (1.38)$$

$$IS^2_{j,1} = \frac{1}{4}(f_{j+1}-f_{j-1})^2+\frac{13}{12}(f_{j+1}-2f_j+f_{j-1})^2, \quad (1.39)$$

$$IS^2_{j,0} = \frac{1}{4}(3f_j-4f_{j+1}+f_{j+2})^2+\frac{13}{12}(f_j-2f_{j+1}+f_{j+2})^2. \quad (1.40)$$

1.5.6 Cálculo de las derivadas mediante ENO, WENO conservando la monotonía. Algoritmos

Utilizando estos métodos controlamos el orden de aproximación en el cálculo de las derivadas y mantenemos la monotonía. La naturaleza jerárquica del algoritmo ENO presentado en la §1.5.4 nos permite filtrar los valores $\dot{f}_j$ de la siguiente manera: si la aproximación de las derivadas de orden 3 que halla el algoritmo no produce un interpolante que preserve la monotonía de los datos, elegimos la aproximación de orden 2, y así sucesivamente hasta que obtengamos una aproximación que cumpla las condiciones del Teorema 1.1, en un caso que llamaremos restricciones de tipo 1 y las condiciones del Teorema 1.3, que llamaremos restricciones de tipo 2. En este segundo caso debemos hacer una estrategia para la elección de los dos últimos valores de la derivada de forma que alcancen el mayor orden posible. Las aproximaciones ENO de orden 1 siempre satisfacen las condiciones del Teorema 1.1.

Cuando utilizamos la aproximación WENO, primero comprobamos que las derivadas cumplen las condiciones de los teoremas con $r = 2$.

En este caso obtendríamos aproximaciones de $\mathcal{O}(h^4)$ si la función es suave. Si no ocurriese, rebajaríamos a $r = 1$ (orden $\mathcal{O}(h^2)$). En otro caso tomaríamos la aproximación ENO de orden 1 que satisface las condiciones.

Algoritmo 1.4. *Evaluación de $\dot{f}_j$ utilizando restricciones de tipo 1.*

for $j = 1, \ldots, n$
  if $m_{j-1} * m_j < 0$
    $\dot{f}_j = 0$
  else
    $\sigma_j = sign(m_j)$
    $l = 3$ *for ENO or* $l = 2$ *for WENO*
    $\dot{f}_j = \dot{f}_j^{\,l}$ (*being* $\dot{f}_j^{\,l}$ *the ENO /WENO approximation of* $f^{'}(x_j)$)
    $q_j = l - 1$
  while $((\mid \dot{f}_j \mid > 3min\{\mid m_{j-1} \mid, \mid m_j \mid\}$ *or* $sign(\dot{f}_j) \neq \sigma_j)$ *and* $q_j > 0)$
      $\dot{f}_j = \dot{f}_j^{\,q_j}$ (*being* $\dot{f}_j^{\,q_j}$ *the ENO /WENO appr. of* $f^{'}(x_j)$)
      $q_j = q_j - 1$
  end
  endif
end

Para aplicar las restricciones de tipo 2 se necesitan los valores de las derivadas en los puntos x_{j-1} y x_j. Así pues, utilizamos el Algoritmo 1.4 para hallar las aproximaciones a las derivadas. Comprobamos en aquellos nodos donde no hemos obtenido orden máximo (supongamos que hemos obtenido orden t_j en el nodo j) la única condición del Teorema 1.3 que nos falta para las derivadas $\dot{f}_{j-1}, (\dot{f}_j)^s, \dot{f}_{j+1}$ desde que $s = l$, es decir el máximo, hasta que $s = t_j + 1$. Si cumpliesen esta condición varias aproximaciones de distinto orden entonces tomaríamos la de mayor como derivada en el nodo j. Si no la cumpliese ninguna entonces la dejaríamos igual.

Algoritmo 1.5. *Evaluación de $\dot{f}_j$ utilizando restricciones de tipo 2.*

for $j = 1, \ldots, n$
  if $m_{j-1} * m_j < 0$
    $\dot{f}_j = 0$
  else
    $\sigma_j = sign(m_j)$
    $l = 3$ *for ENO or* $l = 2$ *for WENO*
    $\dot{f}_j = \dot{f}_j^{\,l}$ (*being* $\dot{f}_j^{\,l}$ *the ENO /WENO approximation of* $f'(x_j)$)
    $q_j = l - 1$
  while $((\mid \dot{f}_j \mid > 3min\{\mid m_{j-1} \mid, \mid m_j \mid\}$ *or* $sign(\dot{f}_j) \neq \sigma_j)$ *and* $q_j > 0)$
    $\dot{f}_j = \dot{f}_j^{\,q_j}$ (*being* $\dot{f}_j^{\,q_j}$ *the ENO /WENO appr. of* $f'(x_j)$)
    $q_j = q_j - 1$
  end
  endif
end
for $j = 2, \ldots, n-1$
  for $k = 1, \ldots, l-1$
    if $q_j = k$ *and* $\dot{f}_j \neq 0$
      $s = l$
      while $(s > k$ *and* $q_j \neq s)$
        if $(cond(\dot{f}_{j-1}, (\dot{f}_j)^s) > 0$ *and* $cond((\dot{f}_j)^s, \dot{f}_{j+1}) > 0)$
          $q_j = s$
        else
          $s = s - 1$
        endif
      end
    endif
  end
end

donde $cond(\dot{f}_j, \dot{f}_{j+1}) > 0 \equiv$ *condiciones of Teorema 1.3 se cumplen*

Como podemos ver estos algoritmos van reduciendo el orden hasta que se satisface la condición de monotonía a diferencia de los métodos vistos hasta ahora donde si la aproximación obtenida no satisface las condiciones de monotonía se restringe para que las cumpla (perdiendo el orden de aproximación).

Veamos en la siguiente sección algunos experimentos numéricos comparándolos con los métodos vistos en la §1.5.3.

1.5.7 Experimentos numéricos

En esta sección comparamos las reconstrucciones que se obtienen con los métodos de Fritsch-Butland, parabólico, parabólico con filtrado (1.19), ENO con restricciones de tipo 1, ENO con restricciones de tipo 2 y WENO con restricciones de tipo 1 y 2. En el primer experimento utilizamos un conjunto de datos monótono de la Tabla 1.3 [45]:

Nodo	x	$f(x)$
0	7,99	0
1	8,09	$2{,}76429 \cdot 10^{-5}$
2	8,19	$4{,}37498 \cdot 10^{-5}$
3	8,7	0,169183
4	9,2	0,469428
5	10	0,943740
6	12	0,998636
7	15	0,999919
8	20	0,999994

Tabla 1.3: *Datos del primer experimento.*

En este primer experimento obtenemos el mismo resultado utilizando ENO con restricciones de tipo 1 y 2. No obtenemos mayor suavidad al utilizar el método WENO con respecto al método ENO. Incluso podemos ver como el método propuesto por Fritsch y Butland en [44] obtiene una mayor suavidad (al menos visual) en el sexto nodo. El interpolante del método parabólico no conserva la monotonía y su restricción queda demasiado *forzada*.

Veamos en la Tabla 1.4 el orden de cada nodo obtenido con los métodos ENO y WENO. La reducción a segundo orden en el nodo 6 ($x = 12$) proporciona un perfil monótono, aunque algo plano (Figura 1.3).

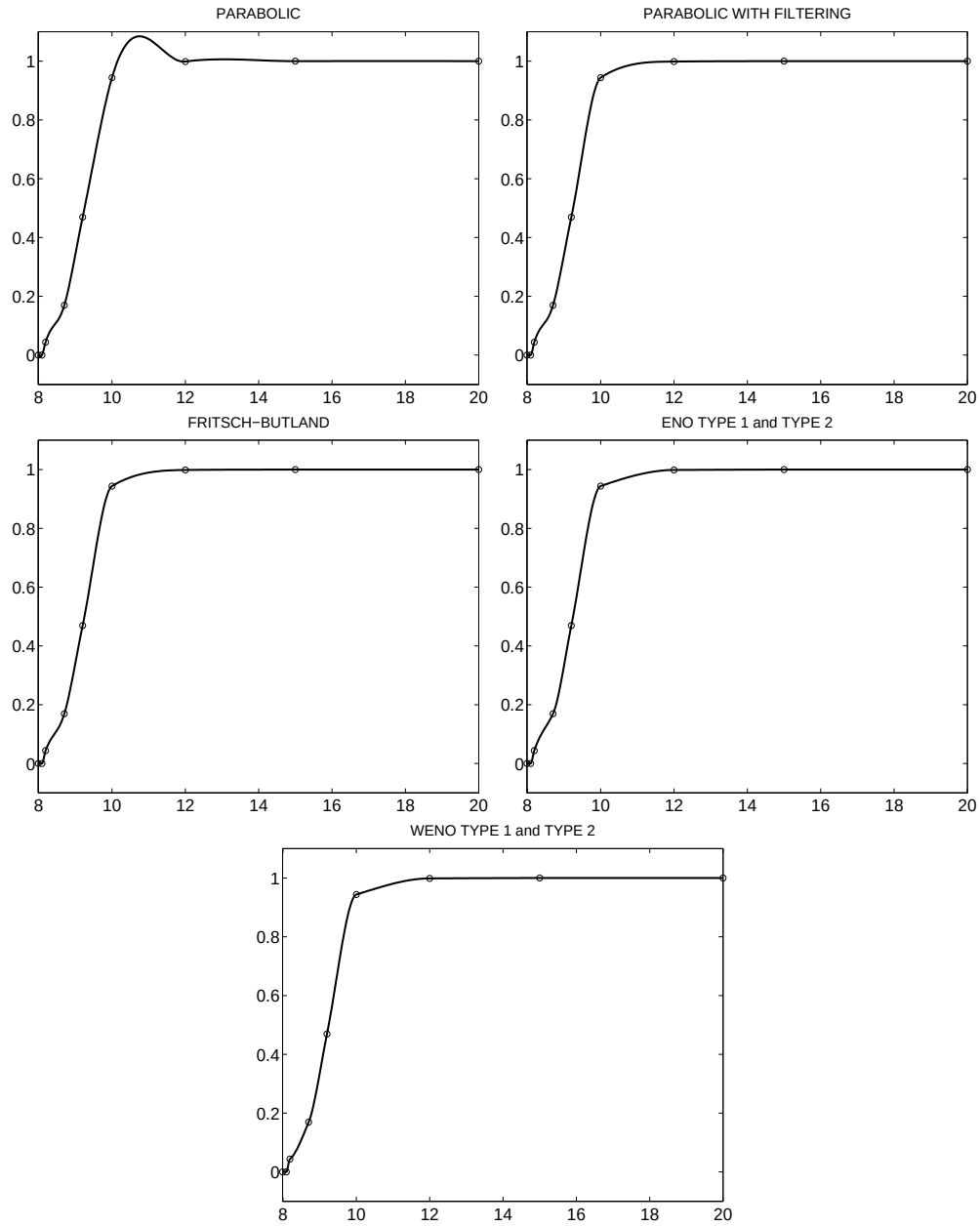


Figura 1.3: *Primer experimento. Datos de la Tabla 1.3 están representados como* o*. La línea continua representa la reconstrucción obtenida a partir de éstos con los distintos métodos.*

Nodo	0	1	2	3	4	5	6	7	8
Orden ENO 1, 2	1	1	3	3	3	3	2	1	1
Orden WENO 1, 2	1	1	4	4	4	4	4	2	1

Tabla 1.4: *Orden de precisión utilizado por los métodos ENO y WENO con restricciones de tipo 1 y 2.*

Nuestro segundo experimento será el ejemplo que utiliza George Wolberg et al. en [93], y cuyos valores podemos ver en la Tabla 1.5.

x	0	1	2	3	4	5	6	7	8	9	10	11
$f(x)$	0	1	4.8	6	8	13	14	15.5	18	19	23	24.1

Tabla 1.5: *Datos del segundo experimento G. Wolberg et al. in [93].*

En el segundo experimento podemos ver (Tabla 1.6) cómo el orden de precisión de las derivadas aumenta utilizando restricciones de tipo 2 en los nodos 1, 2 y 10 dando una mayor suavidad a la función. Por tanto, vemos una pequeña mejora en el orden por haber relajado las condiciones. El coste computacional es mayor (no significativamente).

Nodo	0	1	2	3	4	5	6	7	8	9	10	11
ENO (tipo 1)	1	2	3	3	3	3	3	3	3	2	2	1
ENO (tipo 2)	1	3	3	3	3	3	3	3	3	2	3	1
WENO (tipo 1 y 2)	1	4	4	4	4	4	4	4	4	4	4	1

Tabla 1.6: *Cálculo de los órdenes de precisión en el segundo experimento.*

Podríamos decir que la única ventaja que tenemos al utilizar WENO es que el perfil de la gráfica aparece algo más suave. En WENO también utilizamos restricciones de tipo 2. Utilizar estas condiciones aumenta la suavidad en alguno de los nodos y el orden de precision de las derivadas.

Ahora hacemos el experimento que aparece en el artículo de Fritsch et al. [45], Tabla 1.7.

Como podemos ver en la Figura 1.5 tanto las aproximaciones WENO, como las ENO con restricción de tipo 2 dan una mayor suavidad al perfil de la función entre los nodos 10 y 11 debido a que el orden en el nodo 10 es más alto para el método ENO con restricciones de tipo 2 que para el método ENO con restricciones de tipo 1.

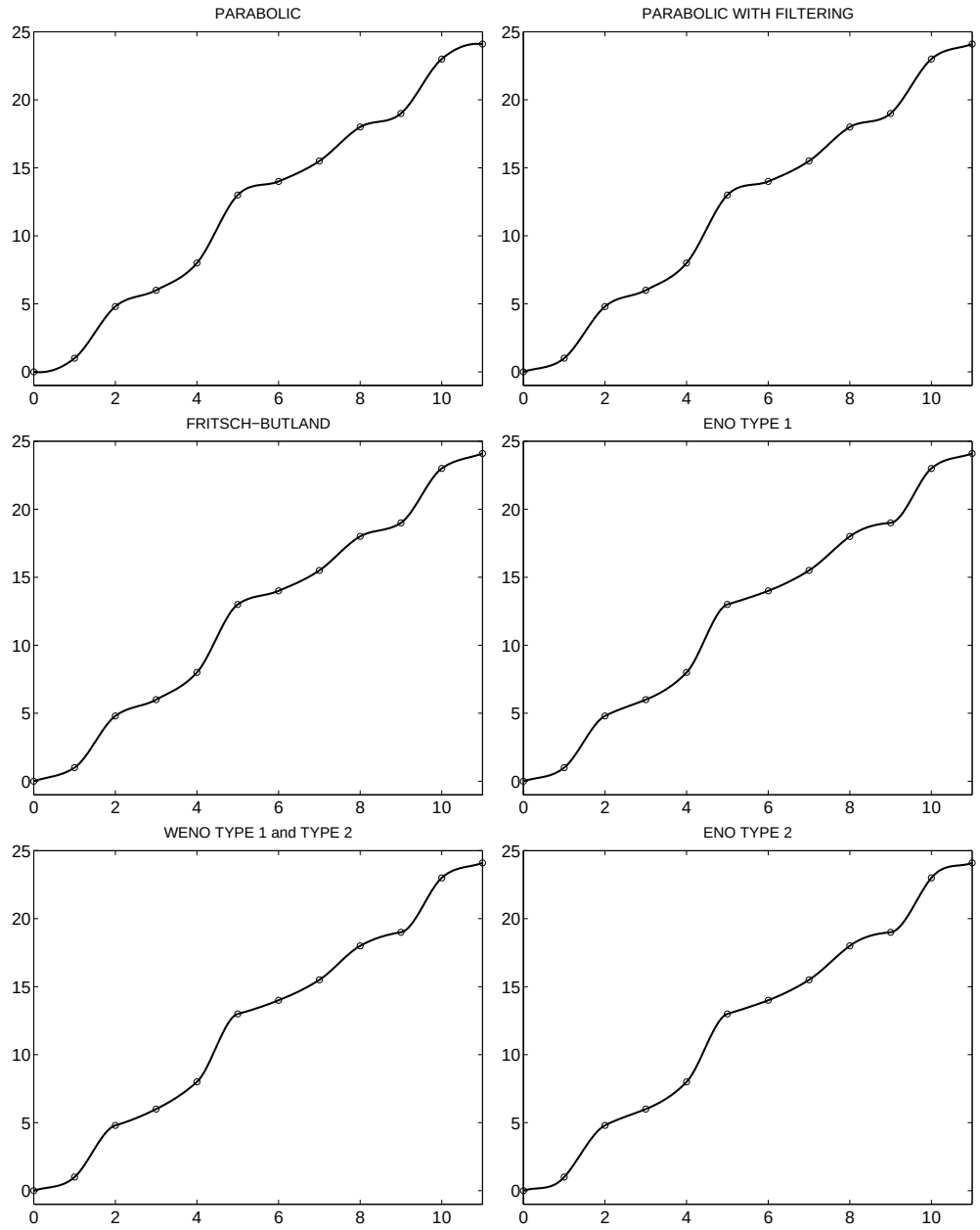


Figura 1.4: *Segundo experimento. Los datos de la Tabla 1.5 están representados como* o*. La línea continua representa la reconstrucción obtenida a partir de éstos con los distintos métodos.*

x	0	2	3	5	6	7	8	9	11	12	14	15
$f(x)$	10	10	10	10	10	10	10	10.5	15	50	60	85

Tabla 1.7: *Datos del tercer experimento.*

Nodo	0	1	2	3	4	5	6	7	8	9	10	11
ENO (tipo 1)	3	3	3	3	3	3	3	3	3	1	3	3
ENO (tipo 2)	3	3	3	3	3	3	3	3	3	3	3	3
WENO (tipo 1)	4	4	4	4	4	4	4	4	4	4	2	4
WENO (tipo 2)	4	4	4	4	4	4	4	4	4	4	4	4

Tabla 1.8: *Cálculo de los órdenes de precisión en el tercer experimento.*

Tomamos el experimento utilizado por Hyman en [63]. Se trata de interpolar la función $f(x) = e^{-x^2}$ en el intervalo $[-1{,}7, 1{,}9]$, con una malla uniforme de n puntos ($\Delta x = \frac{3{,}6}{n-1}$). Representamos la función f con una línea discontinua en las Figuras 1.6 y 1.7. La reconstrucción está marcada por una línea continua. También hemos dibujado (utilizando puntos) la diferencia entre la función original y la reconstrucción. Para que se vea mejor esta diferencia la hemos multiplicado por 10 para el caso $n = 10$ y por 20 para el caso $n = 20$. En este ejemplo como conocemos la función original podemos evaluar el error en norma dos discreta entre ésta y la reconstrucción (Tabla 1.9).

n	FB	Parabólico	Parabólico -MC	ENO (1,2)	WENO
10	$5{,}48 \cdot 10^{-3}$	$2{,}81 \cdot 10^{-3}$	$4{,}49 \cdot 10^{-3}$	$4{,}39 \cdot 10^{-3}$	$4{,}13 \cdot 10^{-3}$
20	$4{,}13 \cdot 10^{-4}$	$2{,}32 \cdot 10^{-4}$	$2{,}40 \cdot 10^{-4}$	$9{,}15 \cdot 10^{-5}$	$8{,}83 \cdot 10^{-5}$

Tabla 1.9: *Norma 2 discreta para el cuarto experimento.*

Para $n = 10, 20$ utilizando los métodos ENO y WENO con restricciones de tipo 1 y 2 obtenemos las mismas derivadas.

En el caso $n = 10$ lo métodos que conservan la monotonía dan como valor máximo al nodo 4 sin ser éste él de la función original y por tanto se comete un error de interpolación mayor, el método parabólico al no conservar la monotonía de los datos obtiene una mejor aproximación a la función f como podemos ver en la Tabla 1.9. En el caso $n = 20$ tenemos un nodo muy cerca del punto $x = 0$ donde la función tiene su máximo. Obtenemos mejores resultados utilizando los métodos ENO y WENO que

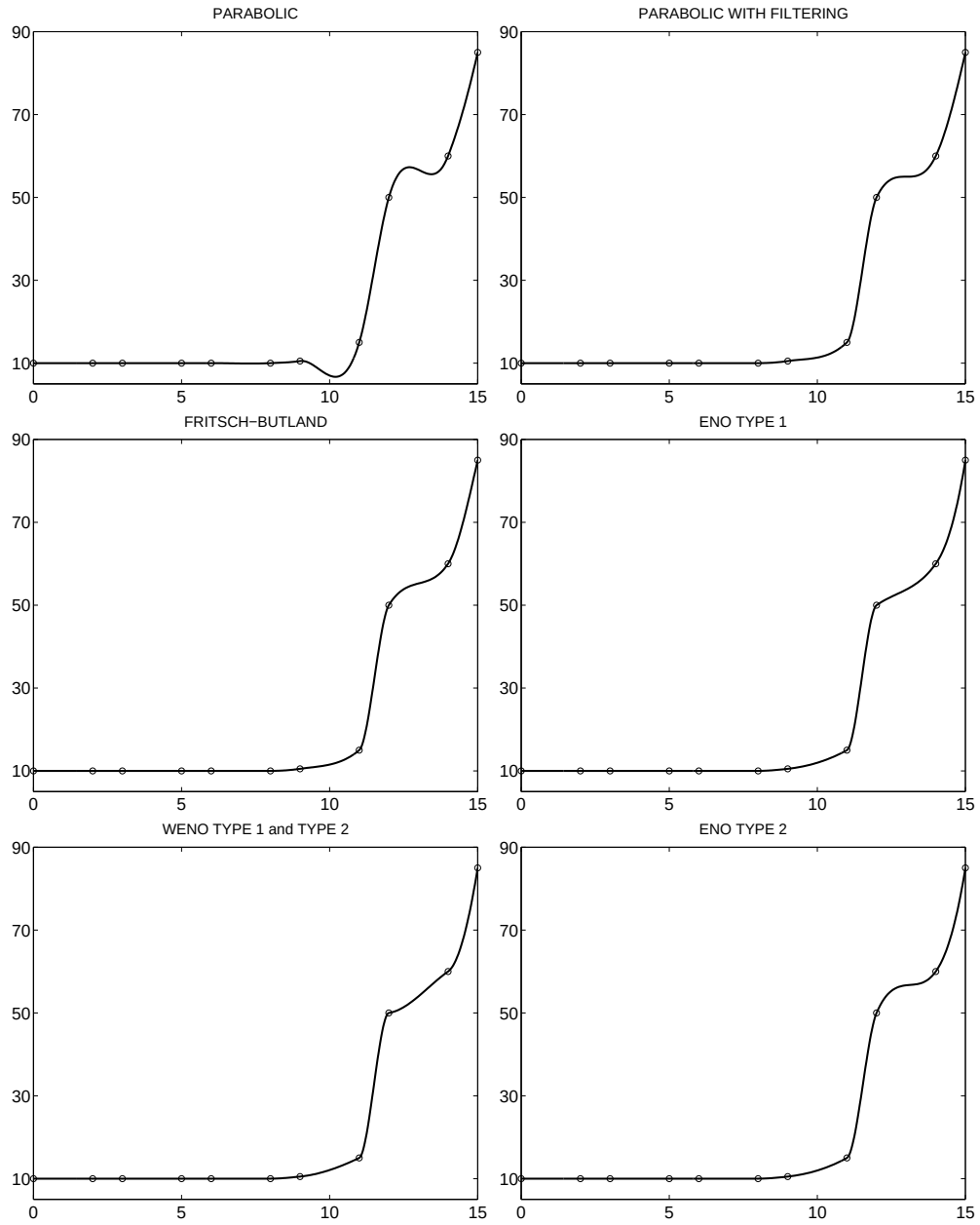


Figura 1.5: *Tercer experimento. Los datos de la Tabla 1.7 están representados como* o*. La línea continua representa la reconstrucción obtenida a partir de éstos con los distintos métodos.*

con el resto de métodos.

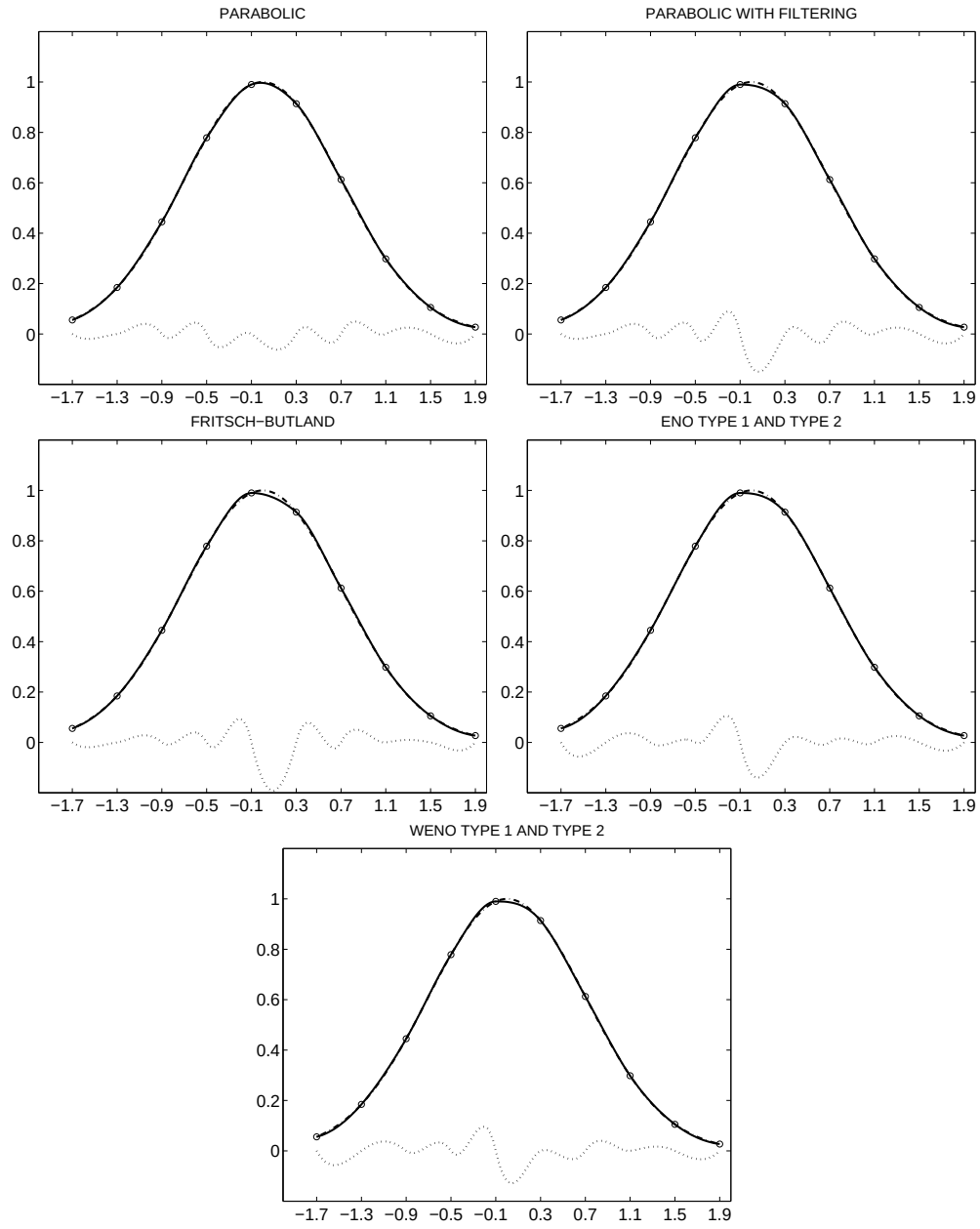


Figura 1.6: *Cuarto experimento con $n = 10$. Línea discontinua: función original. Línea continua: reconstrucción. Puntos: diez veces la diferencia entre la función original y la reconstrucción.*

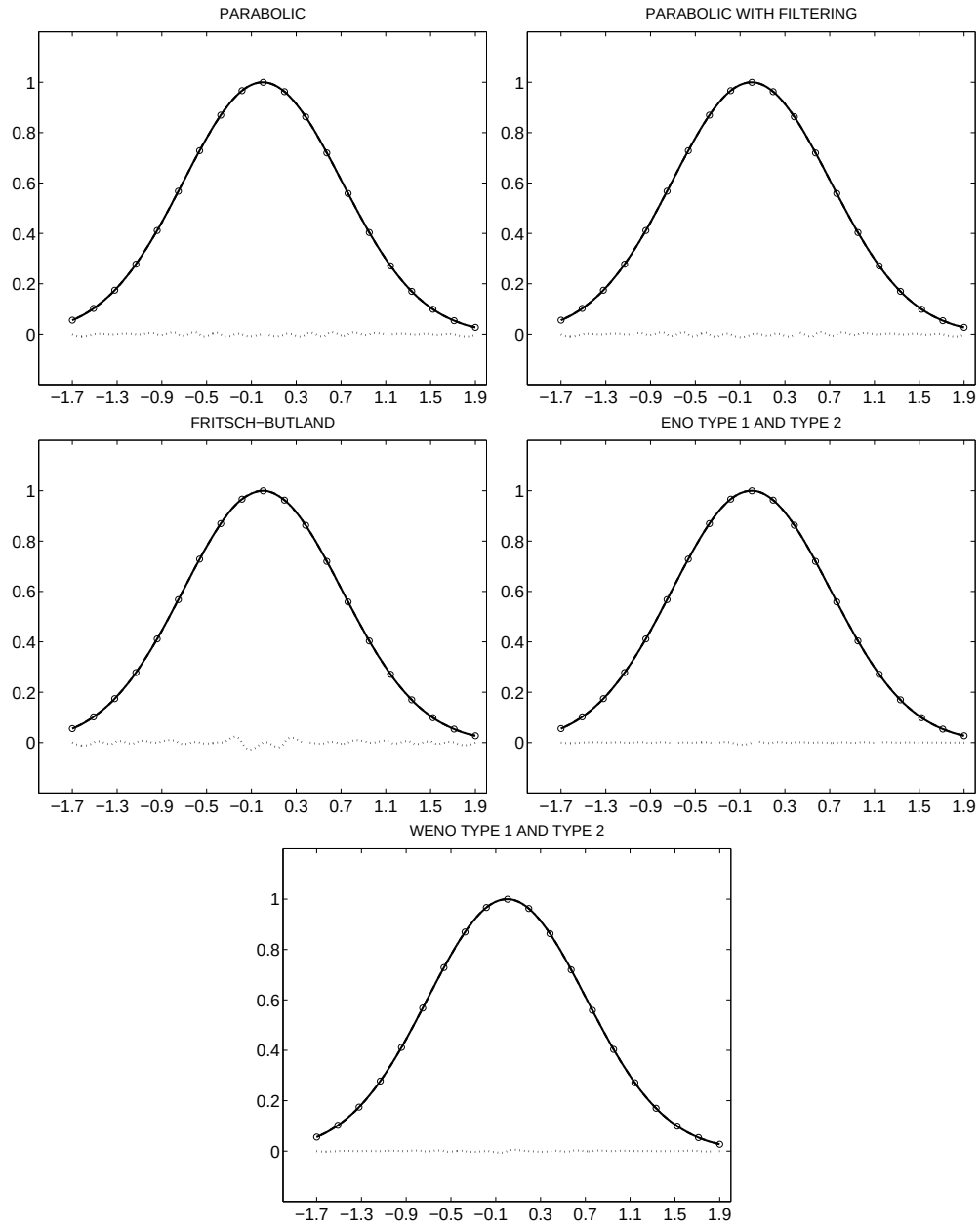


Figura 1.7: *Cuarto experimento con $n = 20$. Línea discontinua: función original. Línea continua: reconstrucción. Puntos: veinte veces la diferencia entre la función original y la reconstrucción.*

Para ver la importancia de utilizar técnicas no lineales para calcu-

lar la derivada terminamos la sección con la función discontinua (1.41) discretizada con $n = 40$ puntos.

$$f(x) = \begin{cases} e^{-x^2} - 0{,}4 & -1{,}7 \leq x < -0{,}5 \\ e^{-x^2} & -0{,}5 \leq x \leq 1{,}9 \end{cases} \tag{1.41}$$

En la Tabla 1.10 podemos ver el error, medido en norma 2, entre la función original y las distintas reconstrucciones. La segunda fila también representa el error de la diferencia pero en este caso esta calculado exceptuando el intervalo, de tamaño $\frac{3{,}6}{40-1}$, que contiene la discontinuidad.

En la Figura 1.8 podemos observar que en los intervalos vecinos del que contiene la singularidad obtenemos mejores resultados utilizando las técnicas propuestas en este trabajo.

n	FB	Parabólico	Parabólico -MC	ENO (1,2)	WENO
40	$3{,}67 \cdot 10^{-2}$	$3{,}65 \cdot 10^{-2}$	$3{,}64 \cdot 10^{-2}$	$3{,}73 \cdot 10^{-2}$	$3{,}73 \cdot 10^{-2}$
40	$2{,}10 \cdot 10^{-3}$	$4{,}47 \cdot 10^{-3}$	$2{,}89 \cdot 10^{-3}$	$2{,}89 \cdot 10^{-4}$	$2{,}92 \cdot 10^{-4}$

Tabla 1.10: *Norma 2 discreta para el quinto experimento.*

Como conclusión señalar que el método ENO ha sido el más fiable, incluso cuando el número de puntos que utilizamos para la interpolación es pequeño. La mejora con restricciones de tipo 2 no es excesivamente significativa como hemos visto en los ejemplos numéricos.

1.6 Conclusiones y futuras líneas de investigación

La utilización de interpolación lineal a trozos en la construcción del operador en los esquemas de multiresolución que veremos en el capítulo 2 ha sido de gran interés por su íntima relación con la teoría de funciones *wavelets*. No solo ha sido de gran interés además ha provocado la posibilidad de construir esquemas de multiresolución no lineales utilizando los diversos métodos, entre otros ENO y WENO que explicamos en este capítulo (ver p.ej. [73, 15, 16] y otros muchos). Mostramos otra utilidad de estos esquemas construyendo un interpolante monótono con diversas propiedades:

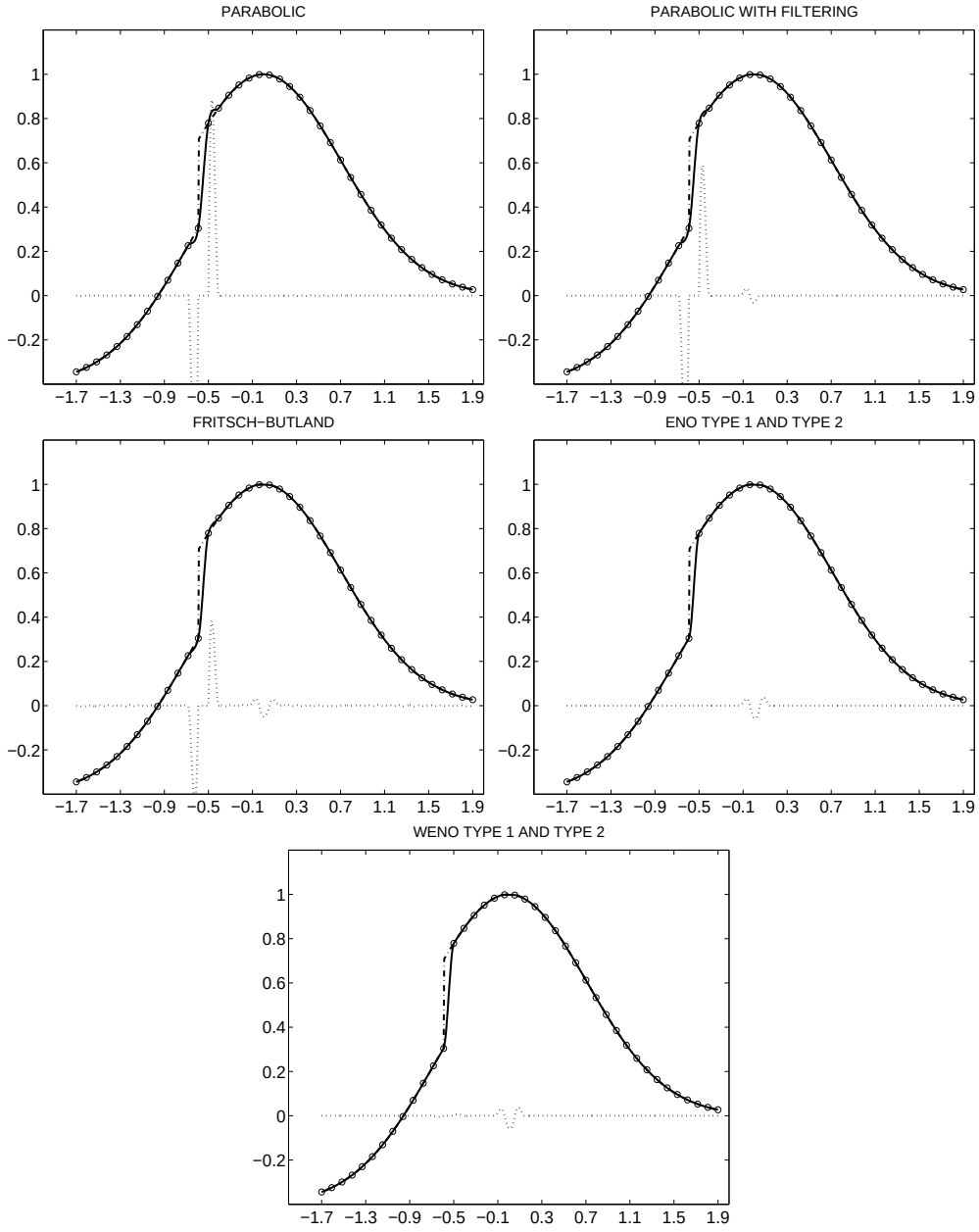


Figura 1.8: *Cuarto experimento con $n = 40$. Línea discontinua: función original. Línea continua: reconstrucción. Puntos: treinta veces la diferencia entre la función original y la reconstrucción (exceptuando el intervalo de tamaño $\frac{3,6}{40.-1}$ que contiene la discontinuidad).*

a) Conserva la monotonía del operador controlando en todo momento el orden de la derivada.

b) Tiene el orden máximo que se puede obtener en cada nodo por su construcción recurrente.

Hemos generalizado los métodos no lineales al cálculo de derivadas con pequeñas modificaciones en los algoritmos y hemos probado algunas de sus propiedades.

La progresión de esta estrategia para conservar monotonía se podría ver en una generalización a varias dimensiones teniendo gran aplicación en campos como la física o en el diseño por ordenador, p. ej. la construcción de la cubierta de un barco o la reconstrucción de elementos faciales por medio de ordenador.

Algunas de las propiedades vistas en este capítulo serán utilizadas para la demostración de las propiedades de esquemas clásicos de multiresolución (capítulo 2) que, a su vez, nos servirán de base para probar ciertas condiciones en los esquemas que diseñaremos usando elementos estadísticos de aprendizaje.

2

Wavelets generalizadas

Las representaciones de multiresolución son herramientas para analizar la información contenida en una señal dada. El desarrollo de la teoría de *wavelets* ([30, 34, 64]) ha producido un gran impacto en varios campos de la ciencia.

Las técnicas de multiescala tienen un importante papel en el análisis numérico.

Burt y Adelson en [23] describen un tipo de algoritmo con estructura de pirámide que trata de calcular una aproximación a una imagen o señal dada utilizando un algoritmo iterativo desde una escala a la siguiente.

Basándose en estos *algoritmos pirámidales* y en la teoría de bases *wavelets* Donoho en [39] propone un esquema de *wavelet interpoladora*. De forma simultánea Harten en [52, 53] formula una teoría general de multiresolución utilizando conocimientos de tres campos diferentes: teoría de bases *wavelets*, solución numérica de ecuaciones en derivadas parciales (EDP) y esquemas de subdivisión dando una visión íntegra de teoría de aproximación al problema. El trabajo de Harten [13] generaliza

las transformaciones *wavelet* y además permite introducir elementos no lineales en su algoritmo (introduce en su esquema interpolación ENO, vista en el capítulo 1). De forma independiente Sweldens desarrolla el esquema *wavelet lifting* en [88] (y otros) que ha inspirado mucha de la literatura presente sobre multiresolución no lineal (el propio esquema de Harten se puede ver como un esquema *wavelet lifting*). En [28], Claypoole et al. utilizan un paso de predicción no lineal para crear una transformación que se adapte al contorno. Heijmans y Goutsias en [59] desarrollan el marco de *wavelets* morfológicas generales para construir *wavelets* no lineales. Otros autores han desarrollado *esquemas pirámidales* no lineales y han estudiado sus propiedades ([5, 4, 16, 31, 46]). Nosotros nos centraremos en explicar la multiresolución à la Harten con detalle sin profundizar en su conexión con la teoría de bases *wavelet* (el lector interesado puede consultar [12, 15, 53]) ni su relación con esquemas de subdivisión ([31, 33]). Introduciremos en los capítulos posteriores una nueva idea dentro del esquema original: teoría estadística de aprendizaje.

2.1 Introducción

Supongamos que obtenemos un conjunto de datos discretos de una señal f (n-dimensional) continua por medio de una función discretización $\mathcal{D}_k$ donde k es el nivel de resolución (a mayor k mayor resolución). Dichos datos están dispuestos en una determinada situación dependiendo de este operador. La multiresolución à la Harten nos permite reordenar la información original obteniendo una disposición más adecuada para su posible almacenamiento comprimiendo así la información (es decir, la podemos almacenar en menos espacio).

Harten formula su problema resolviendo una cuestión principal: ¿Cómo podemos recuperar la señal original f? Es decir, ¿cómo podemos definir un operador reconstrucción $\mathcal{R}_k$, lineal o **no lineal**, que reproduzca exactamente aquellos valores discretos obtenidos por el operador $\mathcal{D}_k$ ($\mathcal{D}_k\mathcal{R}_k = I_{V^k}$) ?.

Combinando estos operadores podemos definir dos nuevos operadores que nos permiten transitar entre dos niveles consecutivos de resolución. Podemos ir de un nivel superior a uno inferior (desde k a $(k-1)$) utilizando el operador decimación $\mathcal{D}_k^{k-1} := \mathcal{D}_{k-1}\mathcal{R}_k$ y desde el nivel $(k-1)$ al k utilizando el operador predicción $\mathcal{P}_{k-1}^k := \mathcal{D}_k\mathcal{R}_{k-1}$. Harten en [52, 53] es-

tudió las propiedades que deben de cumplir estos operadores para poder definir un esquema de multiresolución. En este capítulo repasaremos dichas propiedades.

Basándonos en teoría estadística podemos construir un operador predicción que no satisface una de dichas propiedades (la consistencia). Introduciremos una **nueva estrategia** de multiresolución que se puede aplicar cuando el operador predicción no es consistente.

2.2

Multiresolución *à la Harten*

En el marco general definido por Harten [52, 53] se construyen dos operadores: discretización $\mathcal{D}_k$ y reconstrucción $\mathcal{R}_k$. Estos operadores van desde un espacio de funciones a un espacio discreto de resolución k y viceversa. El índice k denota el nivel de resolución: un k mayor indica mayor resolución.

Sea $\mathcal{F}$ un espacio de funciones, y sea V^k un espacio vectorial. El operador discretización $\mathcal{D}_k : \mathcal{F} \to V^k$ es un operador lineal que a partir de una función $f \in \mathcal{F}$ obtiene unos valores discretos $f^k = \mathcal{D}_k f$. El operador reconstrucción $\mathcal{R}_k : V^k \to \mathcal{F}$ es un operador que a partir de señales discretas obtiene funciones. La multiresolución *à la Harten* es más general que la teoría de bases *wavelet* ya que este operador puede ser ***no lineal***. Veamos en la siguiente figura un ejemplo de discretización y reconstrucción. Suponemos que $\mathcal{D}_k$ es una discretización por valores puntuales y $\mathcal{R}_k$ es una interpolación por medio de un *spline* cúbico.

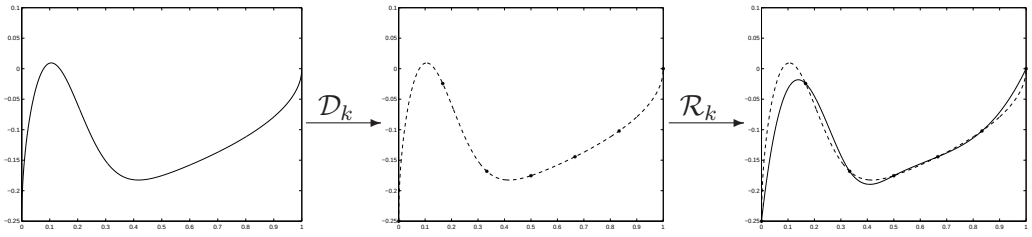


Figura 2.1: *Función* $f(x) = e^{-25x^2}\sqrt{x} - \frac{1}{4}\sqrt{1-x}$. *Operador* $\mathcal{D}_k f$ *en valores puntuales* $(\mathcal{D}_k f)_j = f(j/6)$ *con* $j = 0, \ldots, 6$ *y* $\mathcal{R}_k \mathcal{D}_k f$ *la reconstrucción por medio de interpolación* spline *cúbica.*

La principal relación que deben cumplir estos operadores es la de consistencia,

$$\mathcal{D}_k \mathcal{R}_k = I_{V^k}, \tag{2.1}$$

donde I_{V^k} denota la identidad en V^k. La reconstrucción debe ser consistente con la información discreta de f^k.

Veamos a continuación la definición de discretización encadenada que es otra de las propiedades que debe cumplir el operador discretización para obtener un esquema de multiresolución.

Definición 2.1 (Discretización encadenada)**.** *Sea $\{\mathcal{D}_k\}$ una secuencia de operadores discretización*

$$\mathcal{D}_k : \mathcal{F} \to V^k = \mathcal{D}_k(\mathcal{F}). \tag{2.2}$$

Decimos que la secuencia $\{\mathcal{D}_k\}$ es encadenada si para todo k,

$$\mathcal{D}_k f = 0 \to \mathcal{D}_{k-1} f = 0. \tag{2.3}$$

Esta definición quiere decir que si en un nivel determinado de resolución k no tenemos ninguna información entonces no vamos a encontrar información alguna en niveles menos finos.

Para construir un esquema de multiresolución definimos el operador decimación como $\mathcal{D}_k^{k-1} = \mathcal{D}_{k-1}\mathcal{R}_k$ y el operador predicción como $\mathcal{P}_{k-1}^k = \mathcal{D}_k \mathcal{R}_{k-1}$.

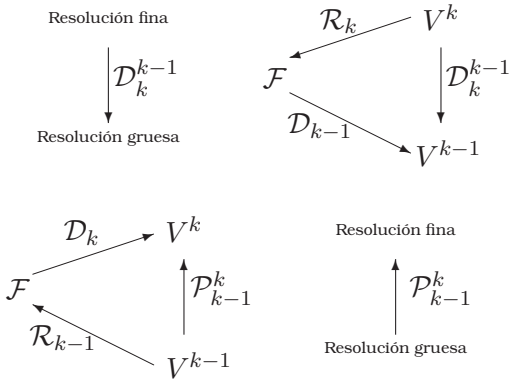


Figura 2.2: *Esquema de los operadores dentro de la multiresolución.*

El operador decimación reduce la señal f^k a f^{k-1} ($f^{k-1} = \mathcal{D}_k^{k-1} f^k$). Es fácil probar que si la secuencia de operadores discretización es encadenada entonces este operador es independiente de la reconstrucción que escojamos (Figura 2.3) [13].

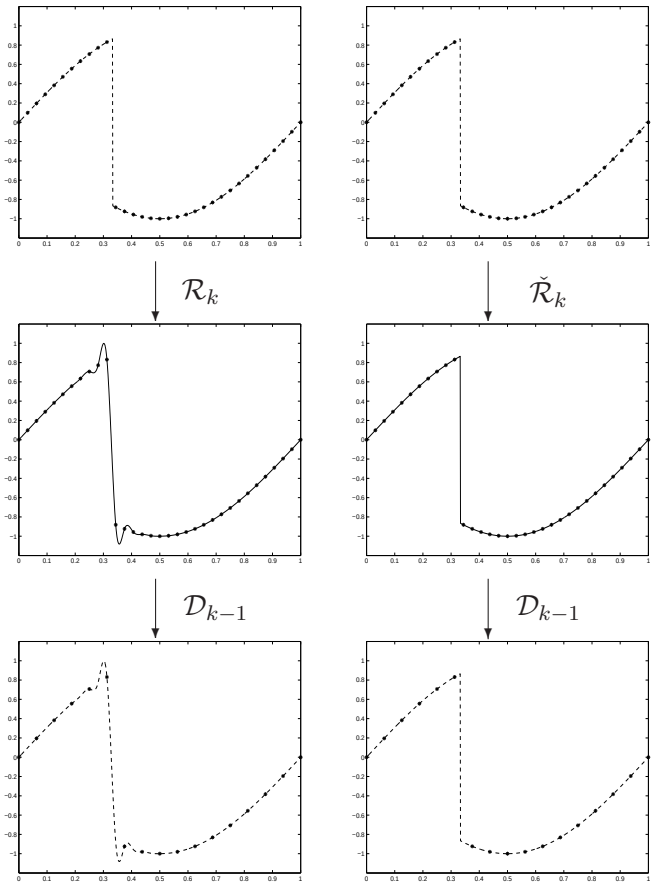


Figura 2.3: *Independencia del operador reconstrucción $\mathcal{R}_k$ en la decimación $\mathcal{D}_k^{k-1}$.*

El operador predicción obtiene una aproximación de f^k a partir de f^{k-1}. Se puede probar que si se cumple la consistencia, (2.1), entonces:

$$\mathcal{D}_k^{k-1}\mathcal{P}_{k-1}^k = I_{V^{k-1}}, \tag{2.4}$$

(y viceversa, siempre y cuando tengamos una discretización encadenada). Al ser $\mathcal{P}_{k-1}^k\mathcal{D}_k^{k-1}f^k$ una aproximación de f^k, podemos definir el error de predicción como:

$$e^k = f^k - \mathcal{P}_{k-1}^k f^{k-1}. \tag{2.5}$$

Los operadores $\mathcal{D}_k^{k-1}$ y $\mathcal{P}_{k-1}^k$ construyen un esquema piramidal de multiresolución [23]. Podemos ver que si $N_k = \dim(V^k)$ entonces e^k tiene

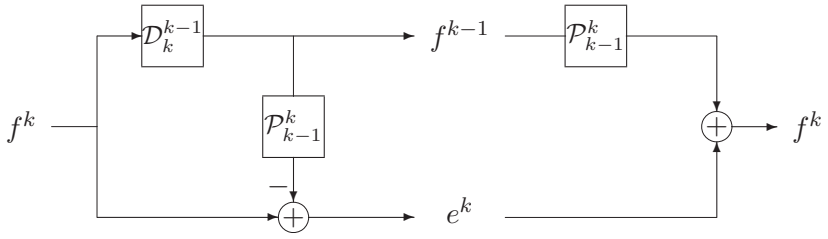


Figura 2.4: *Descomposición en pirámide de la multiresolución à la Harten.*

los mismos elementos que f^k, por tanto en la dupla (f^{k-1}, e^k) hay información redundante que tenemos que eliminar. Para ello, si aplicamos $\mathcal{D}_k^{k-1}$ a (2.5) obtenemos que

$$\mathcal{D}_k^{k-1} e^k = \mathcal{D}_k^{k-1} f^k - \mathcal{D}_k^{k-1} \mathcal{P}_{k-1}^k f^{k-1} = f^{k-1} - f^{k-1} = 0, \tag{2.6}$$

por tanto, $e^k \in \mathcal{N}(\mathcal{D}_k^{k-1}) = \{v \mid v \in V^k, \mathcal{D}_k^{k-1} v = 0\}$. Así, podemos eliminar la información redundante tomando una base $\{\mu_j^k\}$ del espacio $\mathcal{N}(\mathcal{D}_k^{k-1})$ y definiendo una función $G_k : \mathcal{N}(\mathcal{D}_k^{k-1}) \to \mathcal{G}^k$, que asigne a cada elemento $e^k \in \mathcal{N}(\mathcal{D}_k^{k-1})$ las coordenadas d^k en dicha base, i. e. $d^k = G_k e^k$, y sea $\widetilde{G}_k$ la inyección canónica $\mathcal{N}(\mathcal{D}_k^{k-1}) \hookrightarrow V^k$. Entonces $G_k \widetilde{G}_k = I_{\mathcal{G}_k}$ y $\widetilde{G}_k G_k = I_{\mathcal{N}(\mathcal{D}_k^{k-1})}$.

Tenemos, por tanto, una correspondencia uno a uno entre f^k y (f^{k-1}, d^k): dado f^k, obtenemos

$$f^{k-1} = \mathcal{D}_k^{k-1} f^k, \quad d^k = G_k(f^k - \mathcal{P}_{k-1}^k \mathcal{D}_k^{k-1} f^k), \tag{2.7}$$

y, dado $f^{k-1} = \mathcal{D}_k^{k-1} f^k$ y d^k, reconstruimos f^k:

$$\mathcal{P}_{k-1}^k f^{k-1} + \widetilde{G}_k d^k = \mathcal{P}_{k-1}^k \mathcal{D}_k^{k-1} f^k + \widetilde{G}_k G_k (f^k - \mathcal{P}_{k-1}^k \mathcal{D}_k^{k-1} f^k) = f^k. \tag{2.8}$$

Si V^k es de dimensión finita (en esta tesis siempre va a ser así) tenemos $\dim(V^k) = N_k$ y como $\dim(\mathcal{N}(\mathcal{D}_k^{k-1})) = N_k - N_{k-1}$ deducimos que f^k y (f^{k-1}, d^k) tienen exactamente la misma cardinalidad.

Si reiteramos el proceso anterior en el que hemos establecido la equivalencia $f^k \equiv (f^{k-1}, d^k)$ para $k < N$, obtenemos la descomposición multiescala de f^N:

$$f^N \equiv (f^0, d^1, \ldots, d^N).$$

Los algoritmos para obtener la transformación multiescala y su transformación inversa son los siguientes:

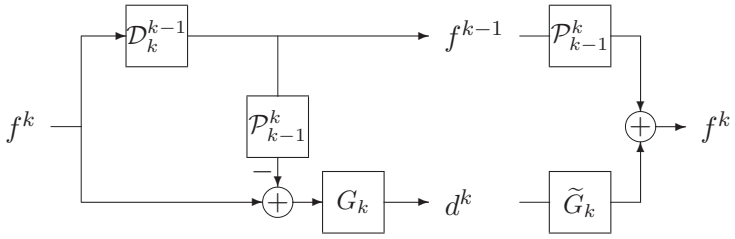


Figura 2.5: *Descomposición sin información redundante utilizando la codificación del error de predicción.*

$$\begin{array}{ccccccc} f^N & \longrightarrow & f^{N-1} & \longrightarrow & f^{N-2} & \longrightarrow & \dots \\ & \searrow & & \searrow & & \searrow & \\ & & d^N & & d^{N-1} & & \dots \end{array}$$

Figura 2.6: *Descomposición multiescala.*

Algoritmo 2.1. *Transformación directa* $f^N \to Mf^N = (f^0, d^1, \dots, d^N)$

for $k = N, \dots, 1$
$f^{k-1} = \mathcal{D}_k^{k-1} f^k$
$d^k = G_k(f^k - \mathcal{P}_{k-1}^k f^{k-1})$
end

Algoritmo 2.2. *Transformación inversa* $Mf^N \to M^{-1}Mf^N$

for $k = 1, \dots, N$
$f^k = \mathcal{P}_{k-1}^k f^{k-1} + \widetilde{G}_k d^k$
end

Los Algoritmos 2.1 y 2.2 tienen la misma estructura que los algoritmos de descomposición y reconstrucción de Mallat ([72]). La descomposición está compuesta por dos filtros: uno de paso bajo, decimación, que es siempre lineal, otro de paso alto, predición. Este último puede ser no lineal.

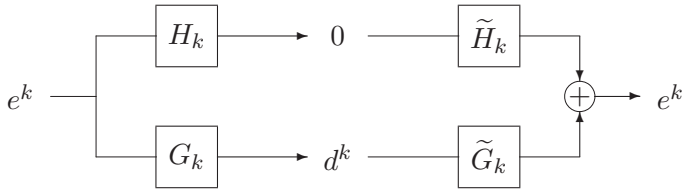


Figura 2.7: *Transformación wavelet para el error de predicción, e^k. El paso bajo de la transformación wavelet en multiresolución à la Harten es el operador decimación, es decir $H_k = \mathcal{D}_k^{k-1}$, y $\widetilde{H}_k$ es su dual en este contexto.*

La multiresolución *à la Harten* puede verse como un paso dentro de un esquema *lifting* desarrollado por Sweldens en [88].

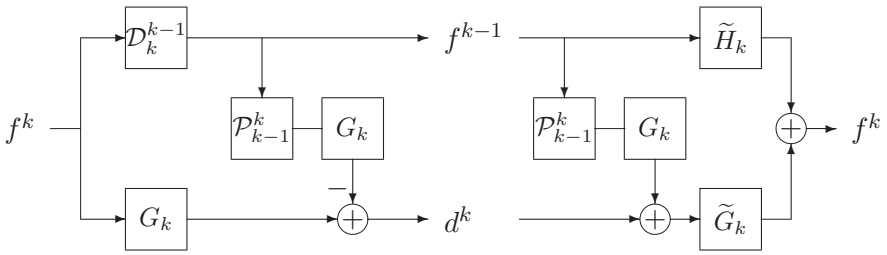


Figura 2.8: *Descomposición sin información redundante vista como un paso de un esquema* lifting*. Generalmente cualquier operador lineal G_k desde V^k a V^{k-1} y $\widetilde{H}_k$ y $\widetilde{G}_k$ desde V^{k-1} a V^k que cumpla $\widetilde{H}_k\mathcal{D}_k^{k-1} + \widetilde{G}_k G_k = I_{V^k}$ constituye un esquema* lifting*.*

2.2.1 ¿Cómo diseñar esquemas de multiresolución?

El primer paso es la elección del operador discretización $\mathcal{D}_k$ (implícitamente estamos eligiendo un operador decimación $\mathcal{D}_k^{k-1}$ ya que si la secuencia es encadenada éste es independiente del operador reconstrucción, $\mathcal{R}_k$). En la próxima sección (§2.2.2) veremos los operadores discretización que utilizaremos durante todo este trabajo (en [46, 53] se pueden ver otras elecciones de operador discretización). La discretización

especifica la naturaleza de los datos f^k (i.e. como fueron generados) y la forma encadenada del operador induce al sentido jerárquico de los distintos niveles de resolución. Elegido este operador, tenemos que tomar una base, $\{\mu_j^k\}$, de su núcleo $\mathcal{N}(\mathcal{D}_k^{k-1})$.

Una vez realizados estos pasos, lo siguiente es construir un operador predicción, $\mathcal{P}_{k-1}^k$, que debe cumplir la relación de consistencia (i.e. ser inverso izquierda del operador $\mathcal{D}_k^{k-1}$). En [16, 13, 46, 53] se construyen operadores predicción basándonos en interpolación polinómica y en técnicas no lineales (ENO y WENO). En esta tesis construimos operadores predicción basándonos en técnicas estadísticas. Éstos no satisfacen la condición de consistencia por lo que introducimos una estrategia que permite diseñar algoritmos de multiresolución en los que el operador predicción no es necesariamente consistente.

A la hora de medir la eficacia del esquema de multiresolución ([53, 13]) analizaremos la capacidad que tiene el operador predicción de reproducir polinomios de distinto grado. Así podemos definir:

Definición 2.2 (Orden del esquema de multiresolución)**.** *Sea p un polinomio de grado r, i.e., $p(x) \in \Pi_1^r(\mathbb{R})$; sea $\{\mathcal{D}_k\}$ una cadena de operadores discretización para cada nivel de resolución k y sea $p^k = \mathcal{D}_k p$. Decimos que el esquema de resolución $\{\mathcal{D}_k^{k-1}, \mathcal{P}_{k-1}^k\}$ tiene orden $r+1$ si y solo si*

$$\mathcal{P}_{k-1}^k \mathcal{D}_k^{k-1} p^k = p^k, \text{ para cualquier nivel de resolución } k.$$

Para aplicar la multiresolución a problemas reales tales como compresión o eliminación de ruido, tenemos que asegurar que las transformaciones directa e inversa son estables con respecto a perturbaciones. Es decir, si tenemos f^N y éstos son reemplazados por una perturbación $\hat{f}^N$ dependiente de ε, entonces queremos controlar la distancia entre los detalles de f^N, d^k, y los detalles de su perturbación, $\hat{d}^k$, es decir queremos que

$$d^k - \hat{d}^k = G_k(I - \mathcal{P}_{k-1}^k \mathcal{D}_k^{k-1}) B_N^k f^N - G_k(I - \mathcal{P}_{k-1}^k \mathcal{D}_k^{k-1}) B_N^k \hat{f}^N, \tag{2.9}$$

con $B_N^k = \mathcal{D}_{k+1}^k \dots \mathcal{D}_N^{N-1}$, esté acotada por una cantidad dependiente de ε y de N. Esta relación (2.9) nos indica que la perturbación en los valores de entrada tienen repercusión en las sucesivas decimaciones $\mathcal{D}_m^{m-1}$ con $m = N, \dots, k+1$, y por tanto en la proyección sobre el núcleo del operador $\mathcal{N}(\mathcal{D}_k^{k-1})$ y su representación en una base. Por tanto, debemos controlar cada decimación sucesiva.

Por otra parte, una vez tenemos los valores $Mf^N = \{f^0, d^1, \dots, d^N\}$, estos son reemplazados por $Q_\varepsilon M f^N = \{\hat{f}^0, \hat{d}^1, \dots, \hat{d}^N\}$ cuando cuantizamos

o truncamos. Debemos controlar que

$$f^N - \hat{f}^N = \sum_{k=0}^{N-1} \left(A_k^N \widetilde{G}_k d^k - A_k^N \widetilde{G}_k \hat{d}^k \right) + (A_0^N f^0 - A_0^N \hat{f}^0) \tag{2.10}$$

con $A_k^N = \mathcal{P}_{N-1}^N \dots \mathcal{P}_k^{k+1}$, esté acotada dependiendo de ε. Definamos, por tanto, qué entendemos por estabilidad de las transformadas directa e inversa.

Definición 2.3 (Estabilidad del esquema de multiresolución)**.** *El algoritmo de descomposición es estable con respecto a la norma* $||\cdot||$ *si* $\exists\, C$ *tal que* $\forall\ j_0,\ \forall\ (f^{j_0}, \hat{f}^{j_0})$ *y sus descomposiciones* $(f^0, d^1, \dots, d^{j_0})$ *y* $(\hat{f}^0, \hat{d}^1, \dots, \hat{d}^{j_0})$ *entonces:*

$$\begin{aligned} &||f^0 - \hat{f}^0|| \leq C||f^{j_0} - \hat{f}^{j_0}|| \\ &||d^k - \hat{d}^k|| \leq C||f^{j_0} - \hat{f}^{j_0}||, \quad \forall\ 1 \leq k \leq j_0. \end{aligned}$$

El algoritmo de reconstrucción es estable con respecto a la norma $||\cdot||$ *si:* $\exists\, C$ *tal que* $\forall\ j_0 > 0$, $\forall\ (f^0, d^1, \dots, d^{j_0}), (\hat{f}^0, \hat{d}^1, \dots, \hat{d}^{j_0})$*:*

$$||f^{j_0} - \hat{f}^{j_0}|| \leq C \sup(||f^{j_0-1} - \hat{f}^{j_0-1}||, ||d^{j_0} - \hat{d}^{j_0}||). \tag{2.11}$$

En el caso en el que el operador predicción, $\mathcal{P}_{k-1}^k$ sea lineal es fácil probar la estabilidad de los algoritmos ([13, 15, 53]). Sin embargo, si éste es no lineal no tenemos garantizada la estabilidad. Para solucionar este problema Aràndiga et al. diseñan una estrategia (*error control*) en la que se modifica la transformación directa y que permite controlar el error ([2]).

Veamos algunos ejemplos de operadores discretización (§2.2.2) y de operadores predicción (§2.2.3).

2.2.2 Operadores discretización

En esta sección vamos a los operadores discretización que utilizaremos durante la tesis. Además definiremos las funciones G_k y $\widetilde{G}_k$ que son necesarias para construir el esquema de multiresolución.

El operador discretización marca la naturaleza de los datos, como fueron generados. Comencemos con el ejemplo más sencillo: **discretización en valores puntuales en** $[0, 1]$ (en un conjunto acotado de mayor dimensión Ω se haría de la misma forma).

Sea X^N una partición uniforme de $[0,1]$:

$$X^N = \{x_j^N\}_{j=0}^{J_N}, \quad x_j^N = jh_N, \quad h_N = 1/J_N, \quad J_N = 2^N J_0,$$

con J_0 un número natural ($J_N < +\infty$). Definimos las mallas de menor resolución $X^k = \{x_j^k\}_{j=0}^{J_k}$, $k = N-1, \ldots, 0$, como

$$x_j^{k-1} = x_{2j}^k, \quad j = 0, \ldots, J_{k-1} := J_k/2.$$

Tomamos como operador discretización el valor de la función en un punto de la malla, es decir,

$$\mathcal{D}_k : \mathcal{F} = \mathcal{B}[0,1] \to V^k \quad f_j^k = (\mathcal{D}_k f)_j = f(x_j^k), \quad 0 \le j \le J_k.$$

Siendo $\mathcal{B}[0,1]$ las funciones reales acotadas en el intervalo $[0,1]$ y V^k el espacio de vectores de dimensión $J_k + 1$.

De $f_j^{k-1} = f(x_j^{k-1}) = f(x_{2j}^k) = f_{2j}^k$, deducimos que el operador decimación es:

$$(\mathcal{D}_k^{k-1} f^k)_j = f_{2j}^k, \quad j = 0, \ldots, J_{k-1}.$$

Como $e^k \in \mathcal{N}(\mathcal{D}_k^{k-1})$ entonces todos los elementos de índice par son 0 ($e_{2j}^k = 0$) y solo es necesario guardar los elementos de índice impar ($d_j^k = e_{2j-1}^k$). Así definimos $(G_k)_{i,j} = \delta_{2i-1,j}$ y su dual es $(\widetilde{G}_k)_{i,j} = \delta_{i,2j-1}$.

Los siguientes ejemplos están basados en la discretización

$$(\mathcal{D}_k f)_j = \int 2^k \phi(2^k x_j^k - 2^k x) f(x) dx = (\phi_k * f)(x_j^k), \text{ con } \phi_k(x) = 2^k \phi(2^k x) \tag{2.12}$$

siendo ϕ una función peso de soporte compacto.

Para la discretización en **medias en celda** ϕ es la función de escala de Haar (Fig. 2.10)

$$\omega_0(x) = \begin{cases} 1, & -\frac{1}{2} \le x \le \frac{1}{2}; \\ 0, & \text{en otro caso.} \end{cases} \tag{2.13}$$

La discretización $(\mathcal{D}_k f)_j$ es la media sobre la celda $c_j^k = (2^{-k}(j-1), 2^{-k} j)$,

$$(\mathcal{D}_k f)_j = 2^k \int_{c_j^k} f(x) dx, \tag{2.14}$$

de ahí el nombre de "medias en celda". La ecuación dilatación de la función de escala de Haar $\omega_0(x) = \omega_0(2x) + \omega_0(2x-1)$ implica que $f_j^{k-1} = \frac{1}{2} f_{2j}^k + \frac{1}{2} f_{2j-1}^k$; por tanto, el operador decimación queda definido como

$$(\mathcal{D}_k^{k-1} f^k)_j = \frac{1}{2} f_{2j}^k + \frac{1}{2} f_{2j-1}^k. \tag{2.15}$$

Para definir los operadores G_k y su dual, $\widetilde{G}_k$ tenemos en cuenta que $\mathcal{D}_k^{k-1}e^k = 0$. Así $e_{2j}^k = -e_{2j-1}^k$

$$d_j^k = (G_k e^k)_j = e_{2j-1}^k, \quad \begin{cases} e_{2j-1}^k = (\widetilde{G}_k d^k)_{2j-1} = d_j^k, \\ e_{2j}^k = (\widetilde{G}_k d^k)_{2j} = -d_j^k. \end{cases} \tag{2.16}$$

Como tercer ejemplo veamos la discretización **en medias *hat***. Tomamos en este caso en (2.12) como ϕ la función *hat* $\omega_0 * \omega_0 = \omega_0^2$ (Fig. 2.10), i. e.

$$\omega_0^2(x) = \begin{cases} 1+x, & -1 \le x \le 0, \\ 1-x, & 0 \le x \le 1, \\ 0, & \text{en otro caso.} \end{cases} \tag{2.17}$$

Como sucedía en la discretización en medias en celda, la ecuación dilatación de ω_0^2:

$$\omega_0^2(x) = \frac{1}{2}\omega_0^2(2x-1) + \omega_0^2(2x) + \frac{1}{2}\omega_0^2(2x+1).$$

determina $\mathcal{D}_k^{k-1}$:

$$(\mathcal{D}_k^{k-1} f^k)_j = \frac{1}{4}f_{2j-1}^k + \frac{1}{2}f_{2j}^k + \frac{1}{4}f_{2j+1}^k.$$

Siguiendo la misma estrategia que para los dos ejemplos anteriores podemos definir G_k y $\widetilde{G}_k$ como:

$$d_j^k = (G_k e^k)_j = e_{2j-1}^k, \quad \begin{cases} e_{2j-1}^k = (\widetilde{G}_k d^k)_{2j-1} = d_j^k, \\ e_{2j}^k = (\widetilde{G}_k d^k)_{2j} = -\frac{1}{2}(d_j^k + d_{j+1}^k). \end{cases} \tag{2.18}$$

Estos tres ejemplos son miembros de una familia de discretizaciones basados en *splines* (Fig. 2.10). Si tomamos en (2.12) ϕ como:

$$\omega_0^{n+1} = \omega_0^n * \chi_{[-\frac{1}{2},\frac{1}{2}]},\ n \ge 1,\ \omega_0^0 = \delta, \tag{2.19}$$

entonces es fácil ver que los coeficientes de la ecuación de dilatación, $\{\alpha_l^n\}$, siguen la relación recursiva

$$\alpha_l^{n+1} = \frac{1}{2}(\alpha_l^n + \alpha_{l-1}^n),\ \alpha_l^0 = \delta_{l,0}. \tag{2.20}$$

Para $n = 0$ tenemos el ejemplo de valores puntuales, $n = 1$ medias en celda y $n = 2$ medias *hat*, ver Figura 2.9.

Getreuer y Meyer, en [46], estudian otros ejemplos de operador discretización basándose en los filtros de base *wavelet*.

En la siguiente sección vamos a definir operadores predicción basándonos en interpolación y repasaremos una nueva técnica basada en medias poliharmónicas ([4, 5]).

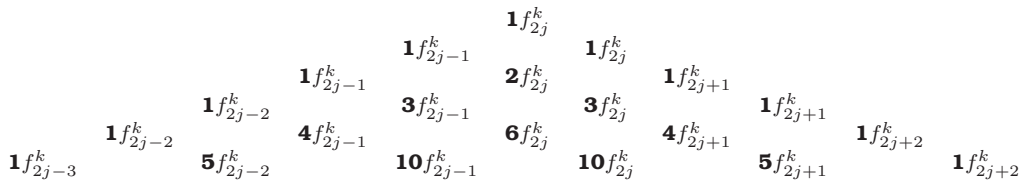


Figura 2.9: *Cada línea* $n = 0, \ldots, 5$*, representa la decimación,* $2^n(\mathcal{D}_k^{k-1} f^k)_j$*, siendo* $n = 0$ *valores puntuales,* $n = 1$ *medias en celda y* $n = 2$ *medias* hat*. Se observa que obtenemos el triángulo de Tartaglia, esto es debido a que estos valores son los coeficientes del binomio de Newton de un polinomio* $(\frac{1+z^{-1}}{2})^n$ *y la decimación puede verse como el paso bajo de un filtro de respuesta de impulso finito (FIR) que se corresponde con este polinomio, para más detalle ver [46].*

2.2.3 Operadores predicción: lineal vs no lineal

Veamos cómo construir operadores predicción. Para ello fijaremos en primer lugar nuestro operador decimación.

Tomando la discretización como $(\mathcal{D}_k f)_j = f(x_j^k) = f_j^k$, **discretización en valores puntuales**. Teniendo en cuenta que queremos que se satisfaga la propiedad de consistencia:

$$\mathcal{D}_k \mathcal{R}_k f^k = f^k, \quad \forall f^k \in V^k; \tag{2.21}$$

entonces tenemos que $\mathcal{R}_k$ debe de cumplir $(\mathcal{R}_k f^k)(x_j^k) = f_j^k = f(x_j^k)$. Así, $(\mathcal{R}_k f^k)(x)$ debe ser una función que interpole al conjunto $\{f_j^k\}$ en los nodos $\{x_j^k\}$. Por tanto

$$(\mathcal{R}_k f^k)(x) := \mathcal{I}(x; f^k), \tag{2.22}$$

y el operador predicción será

$$(\mathcal{P}_{k-1}^k f^{k-1})_j = (\mathcal{D}_k \mathcal{R}_{k-1} f^{k-1})_j = \mathcal{I}(x_j^k; f^{k-1}).$$

Utilizando la terminología de multimalla, el proceso de predicción utiliza la inyección para aquellos valores x_j^k que pertenecen a X^{k-1} e interpola aquellos que no pertenecen a X^{k-1}.

La transformación directa e inversa para valores puntuales (PV) se indica en los algoritmos siguientes (Alg. 2.3, 2.4):

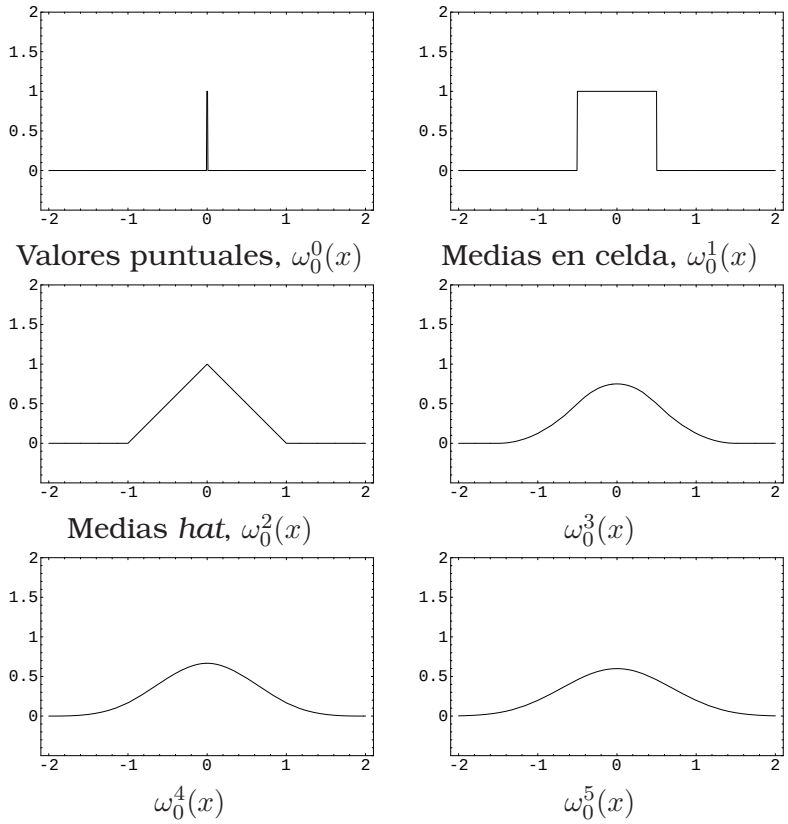


Figura 2.10: *Discretizaciones utilizando la función* $\omega_0^n(x) = \omega_0^{n-1} * \omega_0(x)$ *con* $n = 1, 2, 3, 4, 5$, *siendo* $\omega_0^0(x) = (\delta_{l,0})_{l\in\mathbb{Z}}$ *y* $\omega_0^1(x) = \chi_{[-\frac{1}{2},\frac{1}{2}]}$.

Algoritmo 2.3. *Transformación directa (PV)* $f^N \to Mf^N = (f^0, d^1, \ldots, d^N)$

for $k = N, \ldots, 1$

$$f_j^{k-1} = f_{2j}^k, \qquad 0 \le j \le J_{k-1},$$
$$d_j^k = f_{2j-1}^k - \mathcal{I}(x_{2j-1}^k; f^{k-1}), \qquad 1 \le j \le J_{k-1},$$

end

Algoritmo 2.4. *Transformación inversa (PV)* $Mf^N \to M^{-1}Mf^N$

for $k = 1, \ldots, N$
$f^k_{2j} = f^{k-1}_j, \qquad 0 \le j \le J_{k-1},$
$f^k_{2j-1} = \mathcal{I}(x^k_{2j-1}; f^{k-1}) + d^k_j, \qquad 1 \le j \le J_{k-1},$
end

Si utilizamos la interpolación polinómica a trozos vista en el capítulo 1 con $n+1$ nodos tenemos que si $n = 1$ el operador predicción es

$$(\mathcal{P}^k_{k-1} f^{k-1})_{2j-1} = \frac{1}{2} f^{k-1}_j + \frac{1}{2} f^{k-1}_{j-1}, \tag{2.23}$$

y si $n = 3$ entonces

$$(\mathcal{P}^k_{k-1} f^{k-1})_{2j-1} = -\frac{1}{16} f^{k-1}_{j-2} + \frac{9}{16} f^{k-1}_{j-1} + \frac{9}{16} f^{k-1}_j - \frac{1}{16} f^{k-1}_{j+1}. \tag{2.24}$$

Como los coeficientes d^k_j son los errores de interpolación en los nodos impares de la k-ésima escala entonces, por los resultados vistos en el capítulo 1, si los polinomios utilizados son de grado r tenemos

$$d^k_j = \mathcal{O}(h^{r+1}_{k-1}) \text{ siendo } h_{k-1} = \frac{1}{J_{k-1}} \tag{2.25}$$

en regiones de suavidad, mientras que si existe una singularidad aislada, $x_d \in (x^{k-1}_{j-1}, x^{k-1}_j)$ tal que $[f^{(p)}]_{x_d} = \mathcal{O}(1)$, $(p \le r)$ se tiene

$$d^k_j = \mathcal{O}(h^p_{k-1}). \tag{2.26}$$

Como ya mencionamos en el capítulo 1, las técnicas de interpolación que no dependen de los datos iniciales pierden parte de su exactitud en las regiones próximas a la singularidad. Además, si interpolamos en más puntos, agrandamos la región de interpolación y, por tanto, es más probable que crucemos una discontinuidad. Para evitar esto, se han propuesto técnicas no lineales de interpolación como la técnica ENO vista en la §1.3 y la técnica WENO vista en la §1.4 ([38, 73]).

Si utilizamos la técnica ENO tenemos un esquema de multiresolución que no es estable. En este caso es necesario introducir una estrategia de control del error ([3]).

Recientemente Amat et al. proponen otro tipo de interpolación no lineal basándose en el media poliharmónica (PPH) ([5]) y demuestran su estabilidad ([4]).

La media PPH se define como:

$$pph(x,y) = \left(\frac{sign(x)+sign(y)}{2}\right)\frac{2|x||y|}{|x|+|y|}, \quad \forall\, x,y \in \mathbb{R}\setminus\{0\},$$

donde la función $sign(x)$ es el signo de x; entonces si tomamos el operador predicción dado en (2.24) tenemos:

$$\begin{aligned}(\mathcal{P}_{k-1}^k f^{k-1})_{2j-1} &= -\frac{1}{16}f_{j-2}^{k-1} + \frac{9}{16}f_{j-1}^{k-1} + \frac{9}{16}f_j^{k-1} - \frac{1}{16}f_{j+1}^{k-1} \\ &= \frac{1}{2}(f_{j-1}^k + f_j^k) - \frac{1}{8}\left(\frac{\Delta^2 f_{j+1}^k + \Delta^2 f_j^k}{2}\right)\end{aligned}$$

donde $\Delta^2 f_j^k = f_j^k - 2f_{j-1}^k + f_{j-2}^k$. Si substituimos la media aritmética por media poliharmónica obtenemos el método PPH:

$$(\mathcal{P}_{k-1}^k f^{k-1})_{2j-1} = \frac{1}{2}(f_{j-1}^k + f_j^k) - \frac{1}{8}pph(\Delta^2 f_{j+1}^k, \Delta^2 f_j^k) \tag{2.27}$$

que es no lineal y estable y que, al igual que la interpolación ENO y WENO, no produce efecto *gibbs*.

Podemos construir el operador predicción en los casos de las discretizaciones de **medias en celda** y **medias *hat*** utilizando dos vías: directamente (ver §3.8.1) y vía la función primitiva ([52, 53, 13, 46]). La idea de esta última es reducir la reconstrucción a un problema de interpolación y, por tanto, al caso de valores puntuales.

En el caso de medias en celda la relación entre f y su primitiva F es:

$$F(x) = \int_0^x f(y)dy, \quad f(x) = \frac{d}{dx}F(x). \tag{2.28}$$

entonces $F_j^k = F(x_j^k)$. La secuencia $\{F_j^k\}$ corresponde una discretización en valores puntuales de la función $F(x)$. Así si definimos $F_{-1}^k = 0$ entonces la relación entre $\{f_j^k\}$ y $\{F_j^k\}$ es

$$F_j^k = 2^{-k}\sum_{n=0}^{j} f_n^k, \quad f_j^k = 2^k(F_j^k - F_{j-1}^k), \quad j = 1,\ldots,J_k, \tag{2.29}$$

Si $\mathcal{I}$ es cualquier técnica interpoladora (lineal o no lineal) definimos el operador reconstrucción como:

$$(\mathcal{R}_k f^k)(x) = \frac{d}{dx}\mathcal{I}(x;F^k). \tag{2.30}$$

Con esta definición tenemos asegurada la relación de consistencia. Sea $f^k \in V^k$, entonces

$$(\mathcal{D}_k \mathcal{R}_k f^k)_j = 2^k \int_{x_{j-1}^k}^{x_j^k} \frac{d}{dx} \mathcal{I}(x; F^k) dx = 2^k (F_j^k - F_{j-1}^k) = f_j^k.$$

La derivada (2.30) la hemos de ver en el sentido débil. Como $\mathcal{I}(x; F^k)$ es un polinomio interpolante a trozos, entonces $\mathcal{R}_k f^k$ es continua a trozos.

El operador predicción sería:

$$(\mathcal{P}_{k-1}^k f^{k-1})_j = 2^k \big(\mathcal{I}(x_j^k; F^{k-1}) - \mathcal{I}(x_{j-1}^k; F^{k-1})\big). \tag{2.31}$$

Si utilizamos interpolación polinómica de Lagrange a trozos, gracias a la relación (2.29) obtenemos:

$$\begin{cases} (\mathcal{P}_{k-1}^k f^{k-1})_{2j-1} = \frac{1}{8} f_{j-1}^{k-1} + f_j^{k-1} - \frac{1}{8} f_{j+1}^{k-1}, \\ (\mathcal{P}_{k-1}^k f^{k-1})_{2j} = -\frac{1}{8} f_{j-1}^{k-1} + f_j^{k-1} + \frac{1}{8} f_{j+1}^{k-1}. \end{cases} \tag{2.32}$$

Es fácil probar que $d^k(f) = 2^k d^k(F)$. Entonces, si los polinomios $\mathcal{I}(x; F^k)$ son de grado $r+1$, por lo visto en el capítulo 1 y por la relación anterior, tenemos que en las regiones suaves

$$d^k(f) = \mathcal{O}(2^{-(k-1)(r+1)}). \tag{2.33}$$

Si el *stencil* cruza una discontinuidad en $f^{(p)}$ $(p \leq r)$ tenemos que:

$$d^k(f) = \mathcal{O}\big([f^{(p)}]\big) 2^{-(k-1)p}. \tag{2.34}$$

La transformación directa e inversa para medias en celda (CA) sería:

Algoritmo 2.5. *Transformación directa (CA)* $f^N \to Mf^N = (f^0, d^1, \ldots, d^N)$

for $k = N, \ldots, 1$
$f_j^{k-1} = \frac{1}{2}(f_{2j-1}^k + f_{2j}^k), \qquad 1 \leq j \leq J_{k-1},$
$d_j^k = f_{2j-1}^k - (\mathcal{P}_{k-1}^k f^{k-1})_{2j-1}, \qquad 1 \leq j \leq J_{k-1},$
end

Algoritmo 2.6. *Transformación inversa (CA)* $Mf^N \to M^{-1}Mf^N$

for $k = 1, \dots, N$
$f^k_{2j-1} = (\mathcal{P}^k_{k-1} f^{k-1})_{2j-1} + d^k_j, \qquad 1 \le j \le J_{k-1},$
$f^k_{2j} = 2f^{k-1}_j - f^k_{2j-1}, \qquad 1 \le j \le J_{k-1},$
end

Veamos, por último, el ejemplo de **medias hat**. Tal y como se prueba en [15], a partir de la relación entre la función f y su segunda primitiva:

$$H(x) = \int_0^x \int_0^y f(z)dzdy, \quad f(x) = \frac{d^2}{dx^2} H(x), \tag{2.35}$$

con $H^k_j = 0$ para $j < 0$, tenemos una biyección entre f^k y H^k:

$$H^k_j = 4^{-k} \sum_{m=0}^{j-1} \sum_{n=0}^{m} f^k_n, \quad f^k_j = 4^k (H^k_{j-1} - 2H^k_j + H^k_{j+1}), \quad j = 1, \dots, J_k. \tag{2.36}$$

Definimos el operador reconstrucción como,

$$(\mathcal{R}_k f^k)(x) = \frac{d^2}{dx^2} \mathcal{I}(x; H^k). \tag{2.37}$$

(La segunda derivada debe ser interpretada en el sentido débil).

Se puede probar que se satisface la relación de consistencia ([15] para detalles). El operador predicción es

$$(\mathcal{P}^k_{k-1} f^{k-1})_j = 4^k \big(\mathcal{I}(x^k_{j+1}; H^{k-1}) - 2\mathcal{I}(x^k_j; H^{k-1}) + \mathcal{I}(x^k_{j-1}; H^{k-1}) \big). \tag{2.38}$$

Si utilizamos interpolación segmentaria de Lagrange obtenemos para orden $p = 3$

$$\begin{cases} (\mathcal{P}^k_{k-1} f^{k-1})_{2j-1} = \frac{1}{2} f^{k-1}_{j-1} + \frac{1}{2} f^{k-1}_j, \\ (\mathcal{P}^k_{k-1} f^{k-1})_{2j} = -\frac{1}{4} f^{k-1}_{j-1} + \frac{3}{2} f^{k-1}_j - \frac{1}{4} f^{k-1}_{j+1}. \end{cases}$$

Y para orden $p = 5$

$$\begin{cases} (\mathcal{P}^k_{k-1} f^{k-1})_{2j-1} = -\frac{3}{32} f^{k-1}_{j-2} + \frac{19}{32} f^{k-1}_{j-1} + \frac{19}{32} f^{k-1}_j - \frac{3}{32} f^{k-1}_{j+1}, \\ (\mathcal{P}^k_{k-1} f^{k-1})_{2j} = \frac{3}{64} f^{k-1}_{j-2} - \frac{16}{64} f^{k-1}_{j-1} + \frac{90}{64} f^{k-1}_j - \frac{16}{64} f^{k-1}_{j+1} + \frac{3}{64} f^{k-1}_{j+2}. \end{cases}$$

En esta ocasión la relación entre los coeficientes de los detalles es $d_j^k(f) = -2^{2k+1} d_j^k(H)$, $1 \leq j \leq J_{k-1}$ (ver [15]). Si utilizamos para interpolar en H^k un polinomio de grado $r+1$ entonces obtenemos:

$$d^k(f) = \mathcal{O}(2^{-(k-1)(r-1)}) \tag{2.39}$$

en las zonas suaves y si el *stencil* cruza una discontinuidad en $f^{(p)}$ $(p \leq r)$ tenemos que:

$$d^k(f) = \mathcal{O}\big([f^{(p)}]\big) 2^{-(k-1)p}. \tag{2.40}$$

En este caso las transformadas directa e inversa serían

Algoritmo 2.7. *Transformación directa (HAT)* $f^N \to Mf^N = (f^0, d^1, \ldots, d^N)$

for $k = N, \ldots, 1$
$f_j^{k-1} = \frac{1}{4}(f_{2j-1}^k + 2f_{2j}^k + f_{2j+1}^k), \qquad 1 \leq j \leq J_{k-1},$
$d_j^k = f_{2j-1}^k - (\mathcal{P}_{k-1}^k f^{k-1})_{2j-1}, \qquad 1 \leq j \leq J_{k-1},$
end

Algoritmo 2.8. *Transformación inversa (HAT)* $Mf^N \to M^{-1}Mf^N$

for $k = 1, \ldots, N$
$f_{2j-1}^k = (\mathcal{P}_{k-1}^k f^{k-1})_{2j-1} + d_j^k, \qquad 1 \leq j \leq J_{k-1},$
$f_{2j}^k = 2f_j^{k-1} - \frac{1}{2}(f_{2j-1}^k + f_{2j+1}^k), \qquad 1 \leq j \leq J_{k-1},$
end

2.3

Consistencia de la multiresolución en el contexto de medias en celda en 1D. Estrategia (AY).

En esta tesis obtenemos operadores predicción, en el contexto de medias en celda, que no son consistentes. En esta sección vamos a ver como

podemos diseñar algoritmos de multiresolución que utilizan operadores predicción no consistentes.

Supongamos que estamos en dimensión uno y, por tanto,

$$(\mathcal{D}_k^{k-1} f^k)_j = \frac{1}{2}(f_j^k + f_{j-1}^k), \quad 1 \leq j \leq J_k.$$

La primer estrategia, que llamaremos (E0), consiste en predecir el valor en la celda $2j-1$, $(\mathcal{P}_{k-1}^k f^{k-1})_{2j-1}$, y definir el nuevo operador predicción como

$$\begin{aligned}(\check{\mathcal{P}}_{k-1}^k f^{k-1})_{2j-1} &= (\mathcal{P}_{k-1}^k f^{k-1})_{2j-1},\\ (\check{\mathcal{P}}_{k-1}^k f^{k-1})_{2j} &= 2f_j^{k-1} - (\mathcal{P}_{k-1}^k f^{k-1})_{2j-1}.\end{aligned}$$

De esta forma la estrategia (E0) consistiría en utilizar los Algoritmos 2.5 y 2.6 con el nuevo operador predicción, $(\check{\mathcal{P}}_{k-1}^k f^{k-1})$, que es consistente.

Una segunda estrategia consistiría en forzar la consistencia modificando el operador predicción. Esto es, si denotamos

$$\xi_j^{k-1} := \frac{1}{2}\left((\mathcal{P}_{k-1}^k f^{k-1})_{2j-1} + (\mathcal{P}_{k-1}^k f^{k-1})_{2j}\right),$$

y si $\mathcal{P}_{k-1}^k$ es nuestro operador predicción entonces definimos el nuevo operador como:

$$\begin{aligned}(\check{\mathcal{P}}_{k-1}^k f^{k-1})_{2j-1} &= (\mathcal{P}_{k-1}^k f^{k-1})_{2j-1} + f_j^{k-1} - \xi_j^{k-1},\\ (\check{\mathcal{P}}_{k-1}^k f^{k-1})_{2j} &= (\mathcal{P}_{k-1}^k f^{k-1})_{2j} + f_j^{k-1} - \xi_j^{k-1},\end{aligned}$$

que sería consistente ya que los errores

$$\begin{aligned}\check{e}_{2j-1}^k &= f_{2j-1}^k - (\check{\mathcal{P}}_{k-1}^k f^{k-1})_{2j-1},\\ \check{e}_{2j}^k &= f_{2j}^k - (\check{\mathcal{P}}_{k-1}^k f^{k-1})_{2j},\end{aligned}$$

pertenecen al núcleo del operador decimación:

$$\begin{aligned}\check{e}_{2j-1}^k + \check{e}_{2j}^k &= f_{2j-1}^k - (\check{\mathcal{P}}_{k-1}^k f^{k-1})_{2j-1} + f_{2j}^k - (\check{\mathcal{P}}_{k-1}^k f^{k-1})_{2j}\\ &= 2f_j^{k-1} - (\mathcal{P}_{k-1}^k f^{k-1})_{2j-1} - 2f_j^{k-1} + 2\xi_j^{k-1} - (\mathcal{P}_{k-1}^k f^{k-1})_{2j}\\ &= 2\xi_j^{k-1} - \left((\mathcal{P}_{k-1}^k f^{k-1})_{2j-1} + (\mathcal{P}_{k-1}^k f^{k-1})_{2j}\right)\\ &= 0.\end{aligned}$$

A esta estrategia la denotaremos como (E1). Las transformadas directa e inversa quedarían:

Algoritmo 2.9. *Transformación directa (E1)* $f^N \rightarrow Mf^N = (f^0, d^1, \ldots, d^N)$

for $k = N, \ldots, 1$

$$
\begin{aligned}
f_j^{k-1} &= \tfrac{1}{2}(f_{2j-1}^k + f_{2j}^k), && 1 \le j \le J_{k-1},\\
\xi_j^{k-1} &= \tfrac{1}{2}((\mathcal{P}_{k-1}^k f^{k-1})_{2j-1} + (\mathcal{P}_{k-1}^k f^{k-1})_{2j}), && 1 \le j \le J_{k-1},\\
d_j^k &= f_{2j-1}^k - ((\mathcal{P}_{k-1}^k f^{k-1})_{2j-1} + f_j^{k-1} - \xi_j^{k-1}), && 1 \le j \le J_{k-1},
\end{aligned}
$$

end

Algoritmo 2.10. *Transformación inversa (E1)* $(\hat{f}^0, \hat{d}^1, \ldots, \hat{d}^N) \rightarrow M^{-1}(\hat{f}^0, \hat{d}^1, \ldots, \hat{d}^N)$

for $k = 1, \ldots, N$

$$
\begin{aligned}
\hat{\xi}_j^{k-1} &= \tfrac{1}{2}((\mathcal{P}_{k-1}^k \hat{f}^{k-1})_{2j-1} + (\mathcal{P}_{k-1}^k \hat{f}^{k-1})_{2j}), && 1 \le j \le J_{k-1},\\
\hat{f}_{2j-1}^k &= (\mathcal{P}_{k-1}^k \hat{f}^{k-1})_{2j-1} + \hat{f}_j^{k-1} - \hat{\xi}_j^{k-1} + \hat{d}_j^k, && 1 \le j \le J_{k-1},\\
\hat{f}_{2j}^k &= 2\hat{f}_j^{k-1} - \hat{f}_{2j-1}^k, && 1 \le j \le J_{k-1},
\end{aligned}
$$

end

Vamos a presentar en esta sección una nueva estrategia cuya principal característica es que conserva la naturaleza del operador predicción aunque perdemos la consistencia del esquema de multiresolución. Demostraremos que conserva el orden de aproximación del operador predicción pues tan sólo modifica los errores.

Sea un operador predicción (consistente o no) $\mathcal{P}_{k-1}^k$. Definimos los errores como:

$$
\begin{aligned}
e_{2j-1}^k &= f_{2j-1}^k - (\mathcal{P}_{k-1}^k f^{k-1})_{2j-1},\\
e_{2j}^k &= f_{2j}^k - (\mathcal{P}_{k-1}^k f^{k-1})_{2j},
\end{aligned}
$$

así $(\mathcal{D}_k^{k-1} e^k)_j = f_j^{k-1} - \xi_j^{k-1} =: (d_0^k)_j$.

Sea $(d_1^k)_j = \frac{1}{2}(e_{2j-1}^k - e_{2j}^k)$, entonces es fácil probar que $(f^{k-1}, d_1^k) \equiv f^k$.

En efecto, si tenemos (f^{k-1}, d_1^k),

$$\begin{aligned}
(\mathcal{P}_{k-1}^k f^{k-1})_{2j-1} + (d_0^k)_j + (d_1^k)_j = \\
(\mathcal{P}_{k-1}^k f^{k-1})_{2j-1} + (f_j^{k-1} - \xi_j^{k-1}) + \frac{1}{2}(e_{2j-1}^k - e_{2j}^k) = \\
\frac{1}{2}(f_{2j-1}^k + f_{2j}^k) + \frac{1}{2}(f_{2j-1}^k - f_{2j}^k) = f_{2j-1}^k; \\
(\mathcal{P}_{k-1}^k f^{k-1})_{2j} + (d_0^k)_j - (d_1^k)_j = \\
(\mathcal{P}_{k-1}^k f^{k-1})_{2j} + (f_j^{k-1} - \xi_j^{k-1}) - \frac{1}{2}(e_{2j-1}^k - e_{2j}^k) = \\
\frac{1}{2}(f_{2j-1}^k + f_{2j}^k) - \frac{1}{2}(f_{2j-1}^k - f_{2j}^k) = f_{2j}^k.
\end{aligned}$$

La otra implicación es trivial.

A esta estrategia la llamaremos (AY) [Aràndiga-Yáñez] y los algoritmos de transformación directa e inversa serían

Algoritmo 2.11. *Transformación directa (AY)* $f^N \to Mf^N = (f^0, d^1, \ldots, d^N)$

for $k = N, \ldots, 1$

$f_j^{k-1} = \frac{1}{2}(f_{2j-1}^k + f_{2j}^k)$, $1 \le j \le J_{k-1}$,

$e_{2j-1}^k = f_{2j-1}^k - (\mathcal{P}_{k-1}^k f^{k-1})_{2j-1}$, $1 \le j \le J_{k-1}$,

$e_{2j}^k = f_{2j}^k - (\mathcal{P}_{k-1}^k f^{k-1})_{2j}$, $1 \le j \le J_{k-1}$,

$(d_1)_j^k = \frac{1}{2}(e_{2j-1}^k - e_{2j}^k)$, $1 \le j \le J_{k-1}$,

end

Algoritmo 2.12. *Transformación inversa (AY)* $(\hat{f}^0, \hat{d}_1^1, \ldots, \hat{d}_1^N) \to M^{-1}(\hat{f}^0, \hat{d}_1^1, \ldots, \hat{d}_1^N)$

Sea $\varepsilon = (\varepsilon_k)$ *y* $0 \le \kappa \le 1$
for $k = 1, \ldots, N$
  for $j = 1, \ldots, J_{k-1}$
    $(\tilde{d}_0^k)_j = \hat{f}_j^{k-1} - \frac{1}{2}\big((\mathcal{P}_{k-1}^k \hat{f}^{k-1})_{2j-1} + (\mathcal{P}_{k-1}^k \hat{f}^{k-1})_{2j}\big)$,
    $(\hat{d}_0^k)_j = \mathrm{quad}((\tilde{d}_0^k)_j, \kappa\varepsilon)$,
    $\hat{f}_{2j-1}^k = (\mathcal{P}_{k-1}^k \hat{f}^{k-1})_{2j-1} + (\hat{d}_0^k)_j + (\hat{d}_1^k)_j$,
    $\hat{f}_{2j}^k = (\mathcal{P}_{k-1}^k \hat{f}^{k-1})_{2j} + (\hat{d}_0^k)_j - (\hat{d}_1^k)_j$,
  end
end

Las transformaciones directas de las estrategias (E1) y (AY) son equivalentes ya que los detalles obtenidos en las dos estrategias son iguales:

$$\begin{aligned} d_j^k &= f_{2j-1}^k - ((\mathcal{P}_{k-1}^k f^{k-1})_{2j-1} + f_j^{k-1} - \xi_j^{k-1}) \\ &= f_{2j-1}^k - \big(\frac{1}{2}((\mathcal{P}_{k-1}^k f^{k-1})_{2j-1} - (\mathcal{P}_{k-1}^k f^{k-1})_{2j-1}) + \frac{1}{2}(f_{2j-1}^k + f_{2j}^k)\big) \\ &= \frac{1}{2}((f_{2j-1}^k - (\mathcal{P}_{k-1}^k f^{k-1})_{2j-1}) - (f_{2j-1}^k - (\mathcal{P}_{k-1}^k f^{k-1})_{2j-1})) \\ &= \frac{1}{2}(e_{2j-1}^k - e_{2j-1}^k) = (d_1^k)_j \end{aligned}$$

La diferencia está en la trasformada inversa. Si tenemos en cuenta que $d^k = (d_1^k)$:

$$\text{(E1):}\quad f_{2j-1}^k = \underbrace{(\mathcal{P}_{k-1}^k f^{k-1})_{2j-1} + f_j^{k-1} - \xi_j^{k-1}}_{(\check{\mathcal{P}}_{k-1}^k f^{k-1})_{2j-1}} + \underbrace{d_j^k}_{\check{e}_{2j-1}^k}$$

$$\text{(AY):}\quad f_{2j-1}^k = \underbrace{(\mathcal{P}_{k-1}^k f^{k-1})_{2j-1}}_{(\mathcal{P}_{k-1}^k f^{k-1})_{2j-1}} + \underbrace{f_j^{k-1} - \xi_j^{k-1} + (d_1^k)_j}_{e_{2j-1}^k}$$

$$\text{(E1):}\quad f_{2j}^k = \underbrace{(\mathcal{P}_{k-1}^k f^{k-1})_{2j} + f_j^{k-1} - \xi_j^{k-1}}_{(\check{\mathcal{P}}_{k-1}^k f^{k-1})_{2j}} \underbrace{-d_j^k}_{\check{e}_{2j}^k}$$

$$\text{(AY):}\quad f_{2j}^k = \underbrace{(\mathcal{P}_{k-1}^k f^{k-1})_{2j}}_{(\mathcal{P}_{k-1}^k f^{k-1})_{2j}} + \underbrace{f_j^{k-1} - \xi_j^{k-1} - (d_1^k)_j}_{e_{2j}^k}.$$

Si $\mathcal{P}_{k-1}^k f^{k-1}$ es un operador con ciertas buenas propiedades es esperable que si tenemos d_0^k modificado estas propiedades se conserven mejor con la estrategia (AY) que utilizaría $\mathcal{P}_{k-1}^k f^{k-1}$ para predecir que con la estrategia (E1), que predeciría con el operador $(\tilde{\mathcal{P}}_{k-1}^k f^{k-1})$ que no sabemos sus propiedades.

La estrategia (AY), por tanto, modifica los valores d_0^k obtenidos. Si éstos son cero no modificará nada y obtendremos el algoritmo clásico de multiresolución de medias en celda (Algoritmo 2.5). Así, esta estrategia conserva el orden.

Lema 2.1. *Sea $(\mathcal{D}_k^{k-1}, \mathcal{P}_{k-1}^k)$ un esquema de multiresolución de orden $r+1$ (según la Def. 2.2) no necesariamente consistente entonces utilizando la estrategia (AY) conservamos el orden del esquema.*

Demostración Probaremos el lema en una dimensión, en dos dimensiones es análogo. Sea un polinomio de grado r, $p \in \Pi_1^r(\mathbb{R})$, entonces por hipótesis

$$\mathcal{P}_{k-1}^k \mathcal{D}_k^{k-1} p^k = p^k,$$

para cualquier k. Así $e^k = p^k - \mathcal{P}_{k-1}^k \mathcal{D}_k^{k-1} p^k = 0$ y por tanto cualquier tipo de estrategia que utilicemos para cuantizar los errores no los modifica ya que son 0. La elección de la estrategia no afecta en el orden del esquema. ■

Vamos a probar ahora algunas propiedades de estabilidad.

Sea $X \in \mathbb{R}^n$, $n < \infty$ definimos las normas que utilizaremos para medir el error como:

$$||X||_1 = \frac{1}{n}\sum_{i=1}^{n} |x_i|, \qquad ||X||_2^2 = \frac{1}{n}\sum_{i=1}^{n} x_i^2, \qquad ||X||_\infty = \max_{i=1,\ldots,n} |x_i|. \tag{2.41}$$

Sabiendo que:

$$\begin{aligned} f_{2j-1}^k &= (\mathcal{P}_{k-1}^k f^{k-1})_{2j-1} + (d_0^k)_j + (d_1^k)_j, \\ f_{2j}^k &= (\mathcal{P}_{k-1}^k f^{k-1})_{2j} + (d_0^k)_j - (d_1^k)_j, \end{aligned}$$

Denotaremos como:

$$\begin{aligned} r_{2j-1}^k &= (d_0^k)_j + (d_1^k)_j, \\ r_{2j}^k &= (d_0^k)_j - (d_1^k)_j, \end{aligned}$$

y

$$\begin{aligned}\hat{r}^k_{2j-1} &= (\hat{d}^k_0)_j + (\hat{d}^k_1)_j,\\ \hat{r}^k_{2j} &= (\hat{d}^k_0)_j - (\hat{d}^k_1)_j,\end{aligned}$$

$$\begin{aligned}\hat{f}^k_{2j-1} &= (\mathcal{P}^k_{k-1}\hat{f}^{k-1})_{2j-1} + (\hat{d}^k_0)_j + (\hat{d}^k_1)_j,\\ \hat{f}^k_{2j} &= (\mathcal{P}^k_{k-1}\hat{f}^{k-1})_{2j} + (\hat{d}^k_0)_j - (\hat{d}^k_1)_j,\end{aligned}$$

donde:

$$\begin{aligned}\hat{d}^k_0 &= \mathit{quad}(\tilde{d}^k_0, \kappa\varepsilon),\\ \hat{d}^k_1 &= \mathit{quad}(d^k_1, \varepsilon),\end{aligned} \tag{2.42}$$

siendo:

$$\tilde{d}^k_0 = \hat{f}^{k-1} - \mathcal{D}^{k-1}_k \mathcal{P}^k_{k-1}\hat{f}^{k-1}, \tag{2.43}$$

entonces $f^k = \mathcal{P}^k_{k-1}f^{k-1} + r^k$ y $\hat{f}^k = \mathcal{P}^k_{k-1}\hat{f}^{k-1} + \hat{r}^k$.

Por tanto:

$$||f^k - \hat{f}^k|| \leq ||\mathcal{P}^k_{k-1}f^{k-1} - \mathcal{P}^k_{k-1}\hat{f}^{k-1}|| + ||r^k - \hat{r}^k||.$$

Lema 2.2. *Sean $a, b \in \mathbb{R}$ entonces se cumple que:*

$$|a+b| + |a-b| \leq 2(|a| + |b|) \tag{2.44}$$

$$\text{máx}\{|a+b|, |a-b|\} \leq 2\,\text{máx}\{|a|, |b|\} \tag{2.45}$$

$$(a+b)^2 + (a-b)^2 = 2(a^2 + b^2). \tag{2.46}$$

Proposición 2.1. *Con la notación utilizada durante toda la sección tenemos que:*

$$||r^k - \hat{r}^k||_\infty \leq 2|||d^k - \hat{d}^k|||_\infty,$$
$$||r^k - \hat{r}^k||_1 \leq |||d^k - \hat{d}^k|||_1,$$
$$||r^k - \hat{r}^k||_2^2 = |||d^k - \hat{d}^k|||_2^2,$$

donde:

$$|||d^k - \hat{d}^k|||_2^2 = ||d_0^k - \hat{d}_0^k||_2^2 + ||d_1^k - \hat{d}_1^k||_2^2,$$
$$|||d^k - \hat{d}^k|||_1 = ||d_0^k - \hat{d}_0^k||_1 + ||d_1^k - \hat{d}_1^k||_1,$$
$$|||d^k - \hat{d}^k|||_\infty = \max\{||d_0^k - \hat{d}_0^k||_\infty, ||d_1^k - \hat{d}_1^k||_\infty\}.$$

Demostración Por la definición de las distintas normas (2.41) y por el Lema 2.2 tenemos que

$$\begin{aligned} J_k||r^k - \hat{r}^k||_1 &= \sum_{j=1}^{J_{k-1}} (|r_{2j-1}^k - \hat{r}_{2j-1}^k| + |r_{2j}^k - \hat{r}_{2j}^k|) \\ &= \sum_{j=1}^{J_{k-1}} (|(d_0^k)_j - (\hat{d}_0^k)_j + (d_1^k)_j - (\hat{d}_1^k)_j| + |(d_0^k)_j - (\hat{d}_0^k)_j - \big((d_1^k)_j - (\hat{d}_1^k)_j\big)|) \\ &\leq \sum_{j=1}^{J_{k-1}} (2|(d_0^k)_j - (\hat{d}_0^k)_j| + 2|(d_1^k)_j - (\hat{d}_1^k)_j|) \\ &= 2J_{k-1}||d_0^k - \hat{d}_0^k||_1 + 2J_{k-1}||d_1^k - \hat{d}_1^k||_1 = J_k|||d^k - \hat{d}^k|||_1. \end{aligned}$$

$$\begin{aligned} J_k||r^k - \hat{r}^k||_2^2 &= \sum_{j=1}^{J_{k-1}} ((r_{2j-1}^k - \hat{r}_{2j-1}^k)^2 + (r_{2j}^k - \hat{r}_{2j}^k)^2) \\ &= \sum_{j=1}^{J_{k-1}} (|(d_0^k)_j - (\hat{d}_0^k)_j + (d_1^k)_j - (\hat{d}_1^k)_j|^2 + |(d_0^k)_j - (\hat{d}_0^k)_j - \big((d_1^k)_j - (\hat{d}_1^k)_j\big)|^2) \\ &= \sum_{j=1}^{J_{k-1}} (2((d_0^k)_j - (\hat{d}_0^k)_j)^2 + 2((d_1^k)_j - (\hat{d}_1^k)_j)^2) \\ &= 2J_{k-1}||d_0^k - \hat{d}_0^k||_2^2 + 2J_{k-1}||d_1^k - \hat{d}_1^k||_2^2 = J_k|||d^k - \hat{d}^k|||_2^2. \end{aligned}$$

$$
\begin{aligned}
||r^k - \hat{r}^k||_\infty &= \max_{i=1,\ldots,J_{k-1}} \{|r^k_{2j-1} - \hat{r}^k_{2j-1}| + |r^k_{2j} - \hat{r}^k_{2j}|\} \\
&= \max_{i=1,\ldots,J_{k-1}} \{|(d^k_0)_j - (\hat{d}^k_0)_j + \big((d^k_1)_j - (\hat{d}^k_1)_j\big)|, |(d^k_0)_j - (\hat{d}^k_0)_j - \big((d^k_1)_j - (\hat{d}^k_1)_j\big)|\} \\
&\leq 2 \max_{j=1,\ldots,J_{k-1}} \{|(d^k_0)_j - (\hat{d}^k_0)_j|, |(d^k_1)_j - (\hat{d}^k_1)_j|\} \\
&= 2\,\text{máx}\{||d^k_0 - \hat{d}^k_0||_\infty, ||d^k_1 - \hat{d}^k_1||_\infty\} = 2|||d^k - \hat{d}^k|||_\infty.
\end{aligned}
$$

■

Lema 2.3. *Siguiendo la notación de todo el capítulo, si $\mathcal{P}^k_{k-1}$ es lineal tenemos que:*

$$||d^k_0 - \hat{d}^k_0||_\alpha \leq ||\tilde{d}^k_0 - \hat{d}^k_0||_\alpha + ||I_{V^{k-1}} - \mathcal{D}^{k-1}_k \mathcal{P}^k_{k-1}||_\alpha ||f^{k-1} - \hat{f}^{k-1}||_\alpha \quad (2.47)$$

donde

$$\hat{d}^k_0 = \text{quad}(\tilde{d}^k_0, \kappa\varepsilon), \text{ con } \quad \tilde{d}^k_0 = \hat{f}^{k-1} - \mathcal{D}^{k-1}_k \mathcal{P}^k_{k-1} \hat{f}^{k-1} \text{ y } \alpha = 1, 2, \infty.$$

Demostración

$$
\begin{aligned}
||d^k_0 - \hat{d}^k_0||_\alpha &= ||d^k_0 - \tilde{d}^k_0 + \tilde{d}^k_0 - \hat{d}^k_0||_\alpha \\
&\leq ||\tilde{d}^k_0 - \hat{d}^k_0||_\alpha + ||f^{k-1} - \mathcal{D}^{k-1}_k \mathcal{P}^k_{k-1} f^{k-1} - \hat{f}^{k-1} + \mathcal{D}^{k-1}_k \mathcal{P}^k_{k-1} \hat{f}^{k-1}||_\alpha \\
&= ||\tilde{d}^k_0 - \hat{d}^k_0||_\alpha + ||(I_{V^{k-1}} - \mathcal{D}^{k-1}_k \mathcal{P}^k_{k-1})(f^{k-1} - \hat{f}^{k-1})||_\alpha \\
&\leq ||\tilde{d}^k_0 - \hat{d}^k_0||_\alpha + ||I_{V^{k-1}} - \mathcal{D}^{k-1}_k \mathcal{P}^k_{k-1}||_\alpha ||f^{k-1} - \hat{f}^{k-1}||_\alpha
\end{aligned}
\quad (2.48)
$$

■

Nota 2.1. *Si $\alpha = 2$ entonces:*

$$
\begin{aligned}
||d^k_0 - \hat{d}^k_0||^2_2 &\leq (||\tilde{d}^k_0 - \hat{d}^k_0||_2 + ||I_{V^{k-1}} - \mathcal{D}^{k-1}_k \mathcal{P}^k_{k-1}||_2 ||f^{k-1} - \hat{f}^{k-1}||_2)^2 \\
&\leq 2(||\tilde{d}^k_0 - \hat{d}^k_0||^2_2 + ||I_{V^{k-1}} - \mathcal{D}^{k-1}_k \mathcal{P}^k_{k-1}||^2_2 ||f^{k-1} - \hat{f}^{k-1}||^2_2).
\end{aligned}
\quad (2.49)
$$

Corolario 2.1. *Sea un esquema de multiresolución uno dimensional* $(\mathcal{D}_k^{k-1}, \mathcal{P}_{k-1}^k)_k$ *no necesariamente consistente en el contexto de medias en celda, siendo el operador predicción lineal, es decir:*

$$||\mathcal{P}_{k-1}^k f^{k-1} - \mathcal{P}_{k-1}^k \hat{f}^{k-1}||_\alpha \leq ||\mathcal{P}_{k-1}^k||_\alpha ||f^{k-1} - \hat{f}^{k-1}||_\alpha, \quad \forall\, f^{k-1}, \hat{f}^{k-1} \in V^{k-1}, \tag{2.50}$$

siendo $\alpha = 1, 2, \infty$*, entonces si utilizamos los algoritmos de descomposición y composición siguiendo la estrategia (AY) tenemos que:*
Si $\alpha = 1, 2$ *entonces*

$$\begin{aligned} ||f^L - \hat{f}^L||_\alpha^\alpha \leq & K_\alpha^L ||f^0 - \hat{f}^0||_\alpha^\alpha + \\ & + \sum_{k=0}^{L-1} K_\alpha^k \big(4^{\alpha-1} ||\tilde{d}_0^{L-k} - \hat{d}_0^{L-k}||_\alpha^\alpha + 2^{\alpha-1} ||d_1^{L-k} - \hat{d}_1^{L-k}||_\alpha^\alpha\big) \end{aligned}$$

siendo $K_\alpha = \max_{k=1,\dots,L}(2^{\alpha-1}||\mathcal{P}_{k-1}^k||_\alpha^\alpha + 4^{\alpha-1}||I_{V^{k-1}} - \mathcal{D}_k^{k-1}\mathcal{P}_{k-1}^k||_\alpha^\alpha)$.
Si $\alpha = \infty$ *entonces*

$$||f^L - \hat{f}^L||_\infty \leq K_\infty^L \max_{k=1,\dots,L} \{||f^0 - \hat{f}^0||_\infty, ||d_1^k - \hat{d}_1^k||_\infty, ||\tilde{d}_0^k - \hat{d}_0^k||_\infty\}.$$

siendo $K_\infty = \max_{k=1,\dots,L} ||\mathcal{P}_{k-1}^k||_\infty + 2||I_{V^{k-1}} - \mathcal{D}_k^{k-1}\mathcal{P}_{k-1}^k||_\infty + 2$.

Demostración Sea $\alpha = 1, 2$, entonces tenemos por la hipótesis del teorema, la Proposición 2.1 y el Lema 2.3 que:

$$\begin{aligned} ||f^k - \hat{f}^k||_\alpha^\alpha &= ||\mathcal{P}_{k-1}^k f^{k-1} + r^k - \mathcal{P}_{k-1}^k \hat{f}^{k-1} - \hat{r}^k||_\alpha^\alpha \\ &\leq 2^{\alpha-1}(||\mathcal{P}_{k-1}^k||_\alpha^\alpha ||f^{k-1} - \hat{f}^{k-1}||_\alpha^\alpha + ||r^k - \hat{r}^k||_\alpha^\alpha) \\ &\leq 2^{\alpha-1}||\mathcal{P}_{k-1}^k||_\alpha^\alpha ||f^{k-1} - \hat{f}^{k-1}||_\alpha^\alpha + 2^{\alpha-1}||d_0^k - \hat{d}_0^k||_\alpha^\alpha + 2^{\alpha-1}||d_1^k - \hat{d}_1^k||_\alpha^\alpha \\ &\leq 2^{\alpha-1}||\mathcal{P}_{k-1}^k||_\alpha^\alpha ||f^{k-1} - \hat{f}^{k-1}||_\alpha^\alpha + \\ &\quad + 4^{\alpha-1}||I_{V^{k-1}} - \mathcal{D}_k^{k-1}\mathcal{P}_{k-1}^k||_\alpha^\alpha ||f^{k-1} - \hat{f}^{k-1}||_\alpha^\alpha + \\ &\quad + 4^{\alpha-1}||\tilde{d}_0^k - \hat{d}_0^k||_\alpha^\alpha + 2^{\alpha-1}||d_1^k - \hat{d}_1^k||_\alpha^\alpha \\ &= (2^{\alpha-1}||\mathcal{P}_{k-1}^k||_\alpha^\alpha + 4^{\alpha-1}||I_{V^{k-1}} - \mathcal{D}_k^{k-1}\mathcal{P}_{k-1}^k||_\alpha^\alpha)||f^{k-1} - \hat{f}^{k-1}||_\alpha^\alpha + \\ &\quad + 4^{\alpha-1}||\tilde{d}_0^k - \hat{d}_0^k||_\alpha^\alpha + 2^{\alpha-1}||d_1^k - \hat{d}_1^k||_\alpha^\alpha. \end{aligned} \tag{2.51}$$

Si $\alpha = \infty$, sea $K_\infty = \max_{k=1,\dots,L} ||\mathcal{P}_{k-1}^k||_\infty + 2||I_{V^{k-1}} - \mathcal{D}_k^{k-1}\mathcal{P}_{k-1}^k||_\infty + 2$,

entonces

$$\begin{aligned}||f^k - \hat{f}^k||_\infty &\leq ||\mathcal{P}_{k-1}^k||_\infty ||f^{k-1} - \hat{f}^{k-1}||_\infty + ||r^k - \hat{r}^k||_\infty \\ &\leq ||\mathcal{P}_{k-1}^k||_\infty ||f^{k-1} - \hat{f}^{k-1}||_\infty + 2\,\text{máx}\{||d_0^k - \hat{d}_0^k||_\infty, ||d_1^k - \hat{d}_1^k||_\infty\} \\ &\leq ||\mathcal{P}_{k-1}^k||_\infty ||f^{k-1} - \hat{f}^{k-1}||_\infty + \\ &+ 2\,\text{máx}\{||I_{V^{k-1}} - \mathcal{D}_k^{k-1}\mathcal{P}_{k-1}^k||_\infty ||f^{k-1} - \hat{f}^{k-1}||_\infty + ||\tilde{d}_0^k - \hat{d}_0^k||_\infty, \\ &\qquad\qquad ||d_1^k - \hat{d}_1^k||_\infty\} \\ &\leq K_\infty\,\text{máx}\{||f^{k-1} - \hat{f}^{k-1}||_\infty, ||\tilde{d}_0^k - \hat{d}_0^k||_\infty, ||d_1^k - \hat{d}_1^k||_\infty\}.\end{aligned}$$

■

Corolario 2.2. *Con la notación utilizada durante toda la sección. Sea un esquema de multiresolución uno dimensional $(\mathcal{D}_k^{k-1}, \mathcal{P}_{k-1}^k)_k$ no necesariamente consistente en el contexto de medias en celda, siendo el operador predicción lineal. Si existe $C_\alpha > 0$ con $\alpha = 1, 2, \infty$ tal que*

$$||\hat{d}_0^k - \tilde{d}_0^k||_\alpha \leq C_\alpha(||f^{k-1} - \hat{f}^{k-1}||_\alpha + |||d^k - \hat{d}^k|||_\alpha),\ \forall\, k, \tag{2.52}$$

siendo $\alpha = 1, 2$, y

$$||\hat{d}_0^k - \tilde{d}_0^k||_\infty \leq C_\infty\,\text{máx}\{||f^{k-1} - \hat{f}^{k-1}||_\infty, |||d^k - \hat{d}^k|||_\infty\},\ \forall\, k; \tag{2.53}$$

entonces los algoritmos de descomposición y composición siguiendo la estrategia (AY) son estables.

Demostración La demostración es inmediata utilizando el Corolario 2.1. ■

El Corolario 2.2 se cumple siempre en los casos prácticos.

Para ver con claridad estas diferencias entre las distintas estrategias (E0), (E1), (AY) veamos unos ejemplos.

Consideramos las siguientes funciones:

$$f_1(x) = \begin{cases} sin(2\pi x), & x \leq \frac{1}{3}; \\ -sin(2\pi x), & x > \frac{1}{3}; \end{cases} \qquad f_2(x) = \begin{cases} -8x, & x \leq \frac{1}{3}; \\ 8x + 20, & x > \frac{1}{3}. \end{cases}$$

En los experimentos tomaremos $J_N = 1024$ y $J_0 = 16$, es decir $N = 6$.

Perturbaremos los detalles de la siguiente manera:

$$(\hat{d}_1^k)_j = \begin{cases} (d_1^k)_j, & \textbf{si } |(d_1^k)_j| \geq \varepsilon_k; \\ 0, & \textbf{si } |(d_1^k)_j| < \varepsilon_k, \end{cases}$$

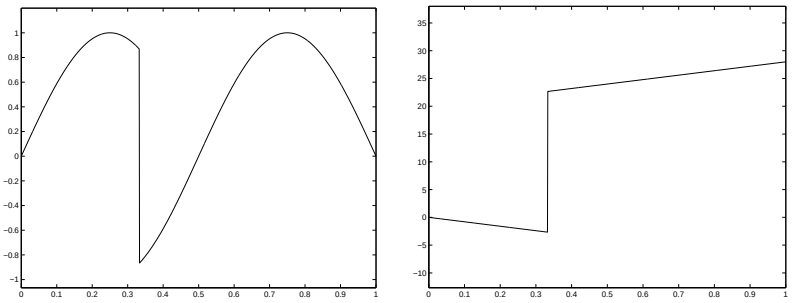


Figura 2.11: *Funciones $f_1(x)$ y $f_2(x)$.*

siendo ε_N dado y $\varepsilon_{k-1} = \varepsilon_k/2$, $1 \leq k \leq N$.

Al aplicar la transformada inversa para la estrategia (AY) (Algoritmo 2.12) modificaremos los valores d_0^k truncando:

$$(\hat{d}_0^k)_j = \begin{cases} (\tilde{d}_0^k)_j, & \text{si } |(\tilde{d}_0^k)_j| \geq \kappa\varepsilon_k; \\ 0, & \text{si } |(\tilde{d}_0^k)_j| < \kappa\varepsilon_k, \end{cases}$$

con $0 \leq \kappa \leq 1$ constante. Si $\kappa = 0$ entonces tenemos la estrategia (E1). De forma experimental hemos visto que obtenemos mejores resultados cuando $\kappa = \frac{1}{4}$.

Para medir el error entre la función original, f^N, y la función reconstruida, $\hat{f}^N$, utilizaremos la norma 2 discreta, es decir:

$$E_2 = \sqrt{\frac{1}{J_N}\sum_j |f_j^N - \hat{f}_j^N|^2},$$

y mediremos el número de elementos no nulos de los errores que almacenamos denotándolos como NNZ.

Vamos a utilizar el siguiente operador predición:

Sean $q_l(x) = a_l + b_l(x)$ y $q_r(x) = a_r + b_r(x)$ tales que $2^k \int_{x_{j-m-1}^k}^{x_{j-m}^k} q_l(x)dx = f_{j-m}^k$, $m = 0, 1$ y $2^k \int_{x_{j+m-1}^k}^{x_{j+m}^k} q_r(x)dx = f_{j+m}^k$, $m = 0, 1$.

Definimos el operador predicción como:

$$\begin{cases} (\mathcal{P}_{k-1}^k f^{k-1})_{2j-1} = 2^{k+1} \int_{x_{2j-2}^k}^{x_{2j-1}^k} q_l(x)dx = \frac{1}{4} f_{j-1}^{k-1} + \frac{3}{4} f_j^{k-1}, \\ (\mathcal{P}_{k-1}^k f^{k-1})_{2j} = 2^{k+1} \int_{x_{2j-1}^k}^{x_{2j}^k} q_r(x)dx = \frac{3}{4} f_j^{k-1} + \frac{1}{4} f_{j+1}^{k-1}. \end{cases} \tag{2.54}$$

Como

$$\xi_j^{k-1} = (\mathcal{D}_k^{k-1}\mathcal{P}_{k-1}^k f^{k-1})_j = \frac{1}{8} f_{j-1}^{k-1} + \frac{3}{4} f_j^{k-1} + \frac{1}{8} f_{j+1}^{k-1},$$

si utilizamos la estrategia (E1) tenemos que

$$\begin{cases} (\check{\mathcal{P}}^k_{k-1} f^{k-1})_{2j-1} = (\mathcal{P}^k_{k-1} f^{k-1})_{2j-1} + (f^{k-1}_j - \xi^{k-1}_j) = \frac{1}{8} f^{k-1}_{j-1} + f^{k-1}_j - \frac{1}{8} f^{k-1}_{j+1}, \\ (\check{\mathcal{P}}^k_{k-1} f^{k-1})_{2j} = (\mathcal{P}^k_{k-1} f^{k-1})_{2j} + (f^{k-1}_j - \xi^{k-1}_j) = -\frac{1}{8} f^{k-1}_{j-1} + f^{k-1}_j + \frac{1}{8} f^{k-1}_{j+1}, \end{cases}$$

que es el esquema interpolatorio de 3 celdas visto en la Ec. (2.32). Este esquema, como veremos en los ejemplos, produce efecto *gibbs*.

Empezamos por la función 1 (Fig 2.11). Como podemos ver en la Figura 2.12 y en la Tabla 2.1 utilizando la estrategia (AY) guardando la misma cantidad de ceros tenemos más o menos el mismo error 2. La diferencia radica en que el esquema (AY) conserva mejor las propiedades del operador predicción (2.54). Cuanto más grande es el valor κ mayor será la adaptación.

	$\varepsilon_N = 0{,}0005$		$\varepsilon_N = 0{,}01$		$\varepsilon_N = 0{,}1$		$\varepsilon_N = 1$	
	NNZ	E_2	NNZ	E_2	NNZ	E_2	NNZ	E_2
(E0)	209	$7{,}45 \cdot 10^{-5}$	55	$1{,}04 \cdot 10^{-3}$	25	0,0041	8	0,0340
(E1)	62	$3{,}17 \cdot 10^{-5}$	33	$2{,}43 \cdot 10^{-4}$	20	0,0017	9	0,0401
(AY)	62	$4{,}80 \cdot 10^{-5}$	33	$3{,}54 \cdot 10^{-4}$	20	0,0018	9	0,0391
	$\varepsilon_N = 2$		$\varepsilon_N = 5$		$\varepsilon_N = 10$		$\varepsilon_N = 15$	
	NNZ	E_2	NNZ	E_2	NNZ	E_2	NNZ	E_2
(E0)	4	0,0707	3	0,0909	0	0,1415	0	0,1415
(E1)	6	0,0644	2	0,0940	1	0,1246	0	0,1595
(AY)	6	0,0632	2	0,0919	1	0,1245	0	0,1674

Tabla 2.1: *Resultados obtenidos para la función $f_1(x)$ utilizando las distintas estrategias (E0), (E1) y (AY) con $\varepsilon_N = 0{,}0005; 0{,}01; 0{,}1; 1; 2; 5; 10$ y 15.*

En la Figura 2.13 vemos que el número de no ceros que obtenemos con las estrategias (AY) y (E1) es el mismo.

La estrategia (AY) es más interesante cuando tenemos una cuantización muy alta ya que en este caso no perdemos la naturaleza del operador predictor y conservamos sus propiedades. Para ver esto nos fijamos en la Figura 2.14 donde se puede ver claramente que cuando la cuantización es más grande, el efecto *gibbs* producido es menor.

En la Figura 2.15 podemos ver que sucede en la discontinuidad al aumentar el nivel de truncamiento (almacenar menos datos) en las estrategias. El operador predicción en (AY), (2.54), no se modifica. Éste es el esquema de subdivisión basado en esquemas *spline* (ver p. ej. el capítulo 2 de [33] para más detalles sobre subdivisión) que no produce efecto *gibbs* pero **no es consistente**. Sin embargo, al utilizar cualquiera

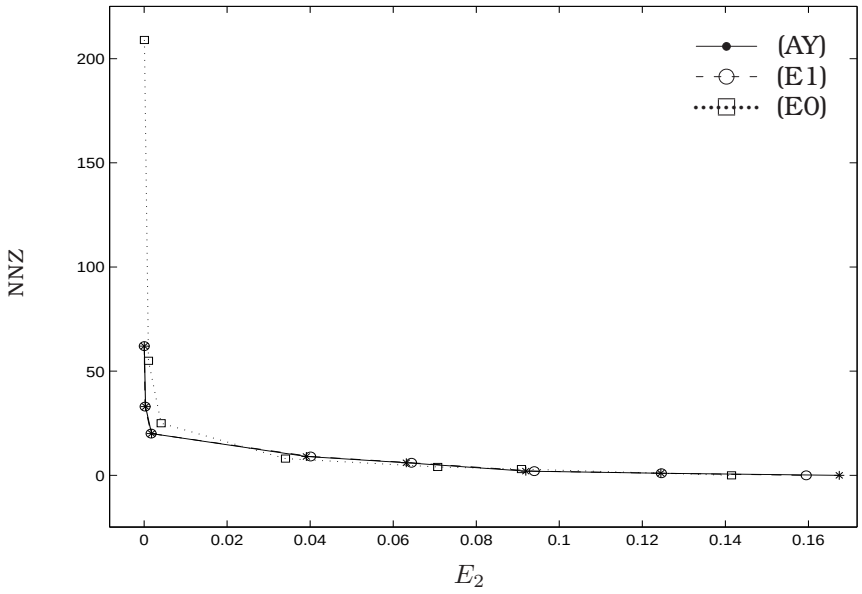


Figura 2.12: *Resultados obtenidos para la función $f_1(x)$ utilizando las distintas estrategias (E0), (E1) y (AY).*

de las otras dos estrategias, (E1) y (E0), el operador se modifica para ser consistente perdiendo su naturaleza y así algunas de las propiedades que tenía (en este ejemplo particular que no producía efecto *gibbs*). Si aumentamos el valor κ en el truncamiento de d_0^k entonces reducimos el efecto *gibbs* pero aumenta la norma 2.

Apliquemos ahora los algoritmos de multiresolución a la función f_2 (Fig 2.11). Vemos que en este segundo ejemplo, al igual que sucedía en el ejemplo anterior, se produce un menor efecto *gibbs* en la discontinuidad debido al predictor elegido. Mostramos el número de elementos no nulos que almacenamos (Figura 2.16) y una figura sinóptica (Figura 2.18) que nos proporciona una visión global de lo que sucede cuando aumentamos el nivel de truncamiento.

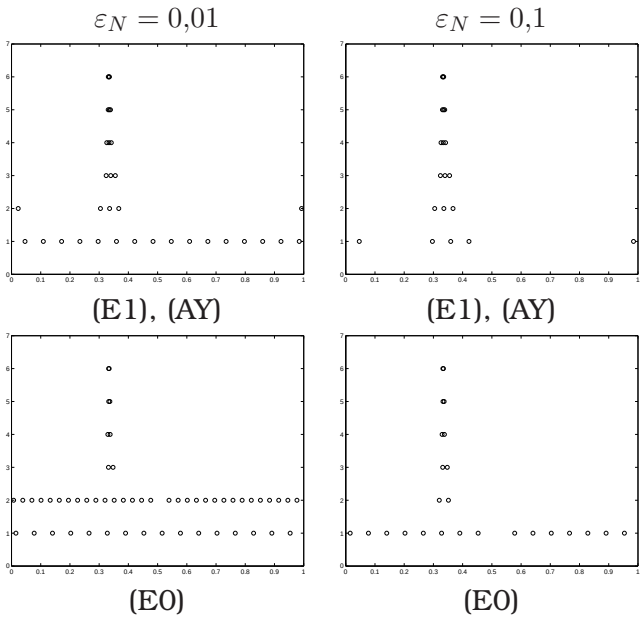


Figura 2.13: *Localización de los detalles d_j^k que se encuentran por encima de la tolerancia ε_k para cada nivel de resolución, con $\varepsilon_N = 0{,}01; 0{,}1$ al aplicar la transformada directa a la función f_1 utilizando las estrategias (E0), (E1) y (AY).*

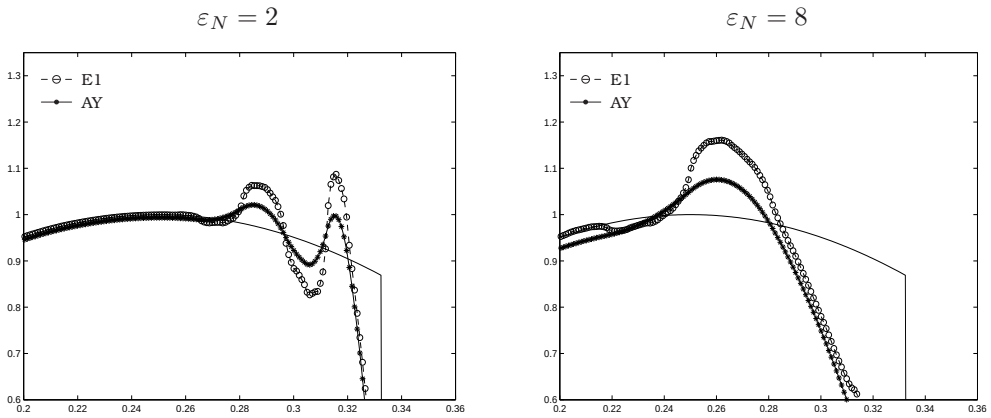


Figura 2.14: *Discontinuidad utilizando estrategias (E1) y (AY) para $\varepsilon_N = 2$ y $\varepsilon_N = 8$.*

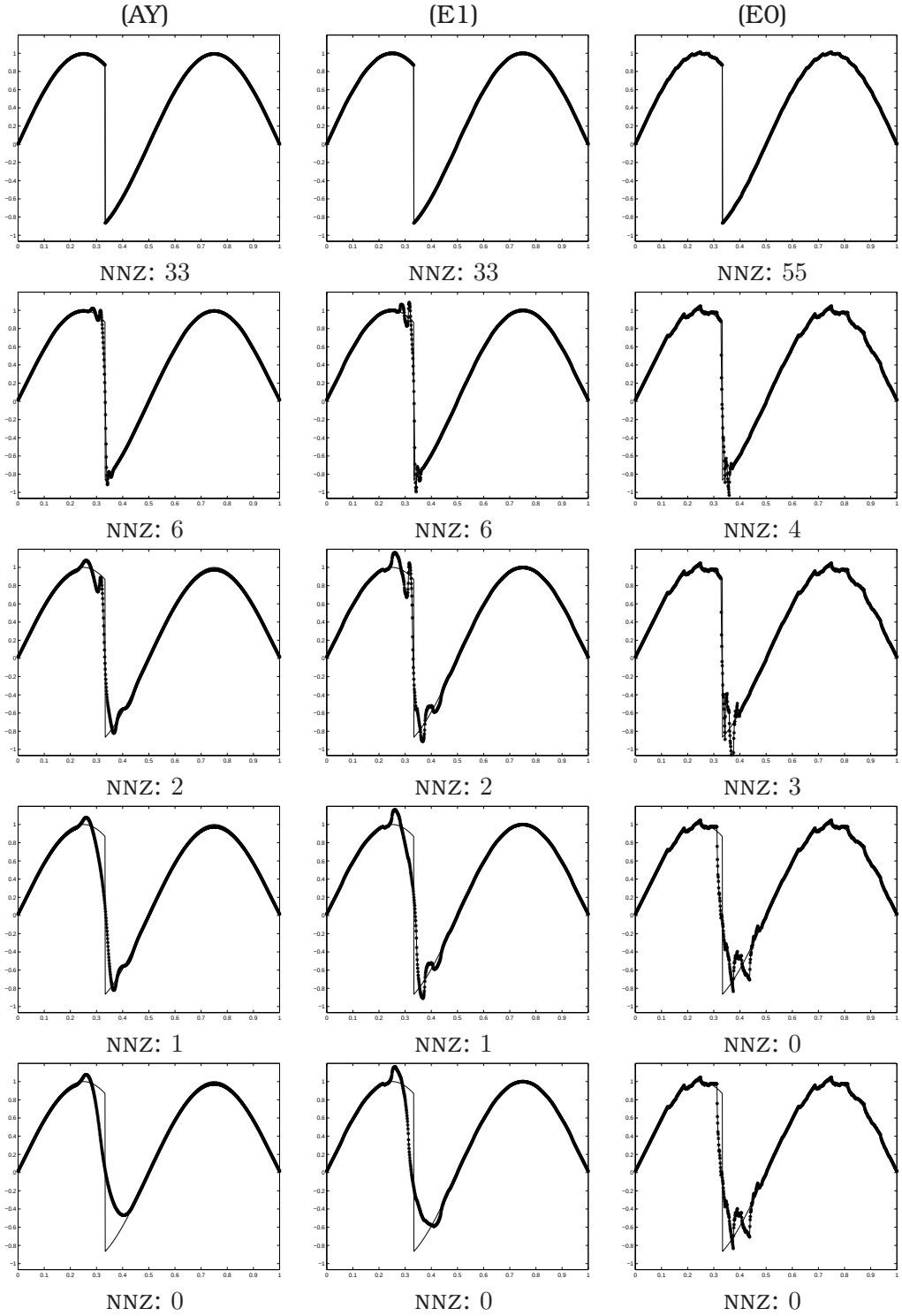


Figura 2.15: *Funciones resultantes realizando multiresolución con la función* f_1 *utilizando las estrategias (E0), (E1) y (AY) para los valores* $\varepsilon_N = 0{,}1; 2; 5; 10; 15$.

	$\varepsilon_N = 2$		$\varepsilon_N = 4$		$\varepsilon_N = 8$		$\varepsilon_N = 16$	
	NNZ	E_2	NNZ	E_2	NNZ	E_2	NNZ	E_2
(E0)	12	0	9	0,1802	9	0,1802	6	0,6491
(E1)	15	0,1082	15	0,1082	10	0,3723	9	0,5838
(AY)	15	0,1011	15	0,1011	10	0,3314	9	0,5837
	$\varepsilon_N = 32$		$\varepsilon_N = 64$		$\varepsilon_N = 128$		$\varepsilon_N = 256$	
	NNZ	E_2	NNZ	E_2	NNZ	E_2	NNZ	E_2
(E0)	4	1,0167	4	1,0167	1	2,1096	0	2,0990
(E1)	6	0,9401	3	1,2686	1	1,8116	0	2,3525
(AY)	6	0,9195	3	1,2474	1	1,8207	0	2,3978

Tabla 2.2: *Resultados obtenidos para la función $f_2(x)$ utilizando las distintas estrategias (E0), (E1) y (AY) con $\varepsilon = 2, 4, 8, 16, 32, 64, 128, 256$.*

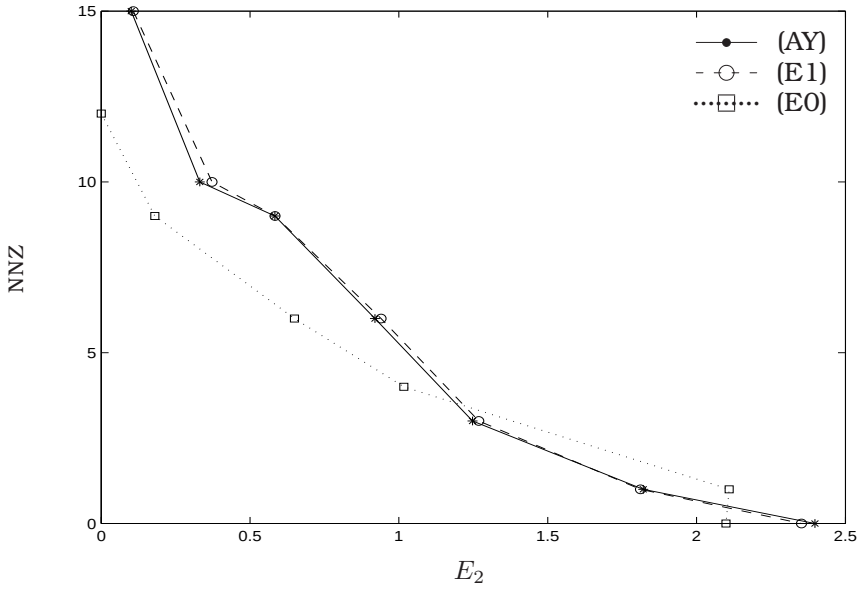


Figura 2.16: *Resultados del segundo experimento.*

2.4

Consistencia de la multiresolución en el contexto de medias en celda en 2D. Estrategia (AY).

En 2D las transformadas directa e inversa para el caso de medias en celda aparecen en los siguientes algoritmos (Alg. 2.13, 2.14):

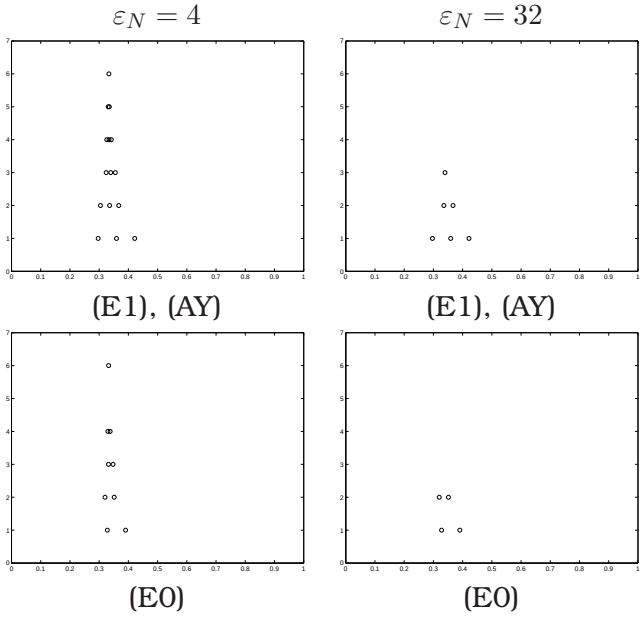


Figura 2.17: *Localización de los detalles d_j^k que se encuentran por encima de la tolerancia ε_k para cada nivel de resolución, con $\varepsilon_N = 4, 32$ al aplicar la transformada directa a la función f_2 utilizando las estrategias (E0), (E1) y (AY).*

Algoritmo 2.13. *Transformación directa CA 2D*

$$f^N \to (f^0, d_1^1, d_2^1, d_3^1, \ldots, d_1^N, d_2^N, d_3^N) :$$

for $k = N, \ldots, 1$
  for $i, j = 1, \ldots, J_{k-1}$
    $f_{i,j}^{k-1} = \frac{1}{4}(f_{2i-1,2j-1}^k + f_{2i,2j-1}^k + f_{2i-1,2j}^k + f_{2i,2j}^k)$,
    $e_{2i-1,2j-1}^k = f_{2i-1,2j-1}^k - (\mathcal{P}_{k-1}^k f^{k-1})_{2i-1,2j-1}$,
    $e_{2i,2j-1}^k = f_{2i,2j-1}^k - (\mathcal{P}_{k-1}^k f^{k-1})_{2i,2j-1}$,
    $e_{2i-1,2j}^k = f_{2i-1,2j}^k - (\mathcal{P}_{k-1}^k f^{k-1})_{2i-1,2j}$,
    $e_{2i,2j}^k = f_{2i,2j}^k - (\mathcal{P}_{k-1}^k f^{k-1})_{2i,2j}$,
    $(d_1^k)_{i,j} = \frac{1}{4}(e_{2i-1,2j-1}^k - e_{2i,2j-1}^k + e_{2i-1,2j}^k - e_{2i,2j}^k)$,
    $(d_2^k)_{i,j} = \frac{1}{4}(e_{2i-1,2j-1}^k + e_{2i,2j-1}^k - e_{2i-1,2j}^k - e_{2i,2j}^k)$,
    $(d_3^k)_{i,j} = \frac{1}{4}(e_{2i-1,2j-1}^k - e_{2i,2j-1}^k - e_{2i-1,2j}^k + e_{2i,2j}^k)$,
  end
end

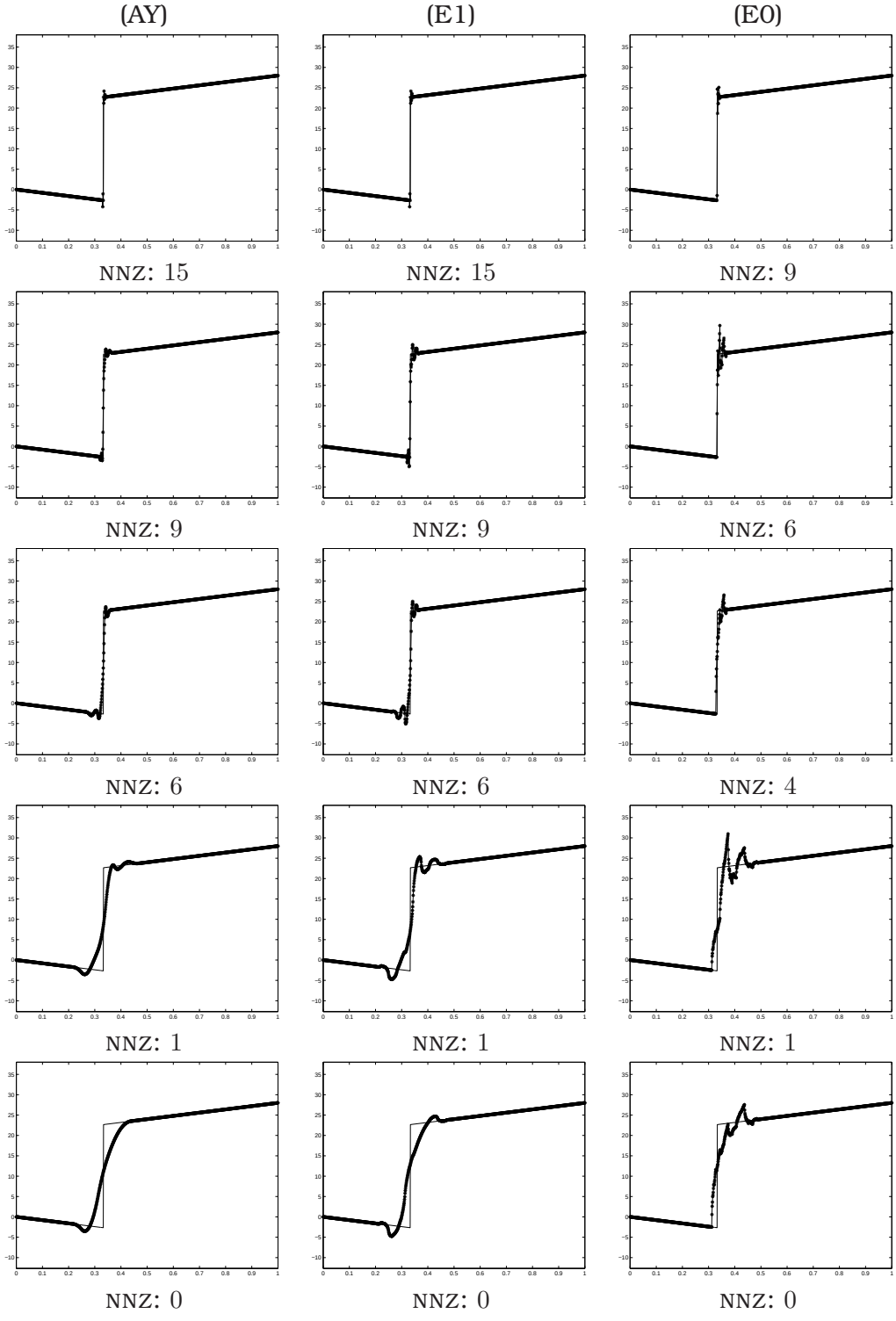


Figura 2.18: *Funciones resultantes realizando multiresolución con la función f_2 utilizando las estrategias (E0), (E1) y (AY) para los valores $\varepsilon_N = 4, 16, 32, 128, 200$. Para el valor $\varepsilon_N = 200$ no guardamos ningún error.*

Algoritmo 2.14. *Transformación inversa*

$(\hat{f}^0, \hat{d}_1^1, \hat{d}_2^1, \hat{d}_3^1, \ldots, \hat{d}_1^N, \hat{d}_2^N, \hat{d}_3^N) \to \hat{f}^N = M^{-1}(\hat{f}^0, \hat{d}_1^1, \hat{d}_2^1, \hat{d}_3^1, \ldots, \hat{d}_1^N, \hat{d}_2^N, \hat{d}_3^N)$

for $k = 1, \ldots, N$
  for $i, j = 1, \ldots, J_{k-1}$
    $\hat{f}^k_{2i-1,2j-1} = (\mathcal{P}^k_{k-1}\hat{f}^{k-1})_{2i-1,2j-1} + (\hat{d}^k_1)_{i,j} + (\hat{d}^k_2)_{i,j} + (\hat{d}^k_{3,\varepsilon})_{i,j}$
    $\hat{f}^k_{2i,2j-1} = (\mathcal{P}^k_{k-1}\hat{f}^{k-1})_{2i-1,2j-1} - (\hat{d}^k_1)_{i,j} + (\hat{d}^k_2)_{i,j} - (\hat{d}^k_3)_{i,j}$
    $\hat{f}^k_{2i-1,2j} = (\mathcal{P}^k_{k-1}\hat{f}^{k-1})_{2i-1,2j-1} + (\hat{d}^k_1)_{i,j} - (\hat{d}^k_2)_{i,j} - (\hat{d}^k_3)_{i,j}$
    $\hat{f}^k_{2i,2j} = (\mathcal{P}^k_{k-1}\hat{f}^{k-1})_{2i,2j} - (\hat{d}^k_1)_{i,j} - (\hat{d}^k_2)_{i,j} + (\hat{d}^k_3)_{i,j}$
  end
end

Si el operador predicción es consistente tenemos que $\mathcal{D}^{k-1}_k \mathcal{P}^k_{k-1} = I_{V^{k-1}}$ y $e^k \in \mathcal{N}(\mathcal{D}^{k-1}_k)$ con lo que

$$e^k_{2i,2j} + e^k_{2i-1,2j} + e^k_{2i,2j-1} + e^k_{2i-1,2j-1} = 0.$$

Esto junto con la definición de d_1, d_2 y d_3 nos da que $e^k \equiv (d^k_1, d^k_2, d^k_3)$ y, por tanto, $f^N \equiv (f^0, d^1_1, d^1_2, d^1_3, \ldots, d^N_1, d^N_2, d^N_3)$

Veamos como podemos construir algoritmos de multiresolución en 2D en el caso en que el operador predicción no sea consistente.

La primera alternativa posible es la más sencilla pero es con la que obtenemos peores resultados. Se trata de calcular los operadores para los píxeles $(2i-1, 2j), (2i, 2j-1), (2i, 2j)$ y definir el operador para el píxel $(2i-1, 2j-1)$ como

$$\begin{aligned}(\check{\mathcal{P}}^k_{k-1} f^{k-1})_{2i-1,2j-1} =& 4f^{k-1}_{i,j} - (\mathcal{P}^k_{k-1} f^{k-1})_{2i-1,2j} \\ & - (\mathcal{P}^k_{k-1} f^{k-1})_{2i,2j-1} - (\mathcal{P}^k_{k-1} f^{k-1})_{2i,2j}.\end{aligned}$$

De esta forma, al igual que en 1D, tendríamos asegurada la consistencia y las transformadas directa e inversa que utilizaríamos serían los Algoritmos 2.13 y 2.14. A esta estrategia la llamaremos (E0).

Otra forma de poder diseñar un algoritmo de multiresolución utilizando operadores predicción no consistentes es modificar las cuatro predicciones para forzar a que lo sean. Esto es, si denotamos

$$\begin{aligned}\xi^{k-1}_{i,j} = \frac{1}{4}\Big(& (\mathcal{P}^k_{k-1} f^{k-1})_{2i-1,2j-1} + (\mathcal{P}^k_{k-1} f^{k-1})_{2i-1,2j} + \\ & + (\mathcal{P}^k_{k-1} f^{k-1})_{2i,2j-1} + (\mathcal{P}^k_{k-1} f^{k-1})_{2i,2j}\Big),\end{aligned} \tag{2.55}$$

definimos el operador predicción como:

$$(\check{\mathcal{P}}_{k-1}^k f^{k-1})_{2i-m_1,2j-m_2} = (\mathcal{P}_{k-1}^k f^{k-1})_{2i-m_1,2j-m_2} + (f_{i,j}^{k-1} - \xi_{i,j}^{k-1}), \qquad (2.56)$$

con $0 \leq m_1, m_2 \leq 1$.

De esta forma el operador $\check{\mathcal{P}}_{k-1}^k$ es consistente y $\mathcal{D}_k^{k-1}\check{e}^k = 0$, siendo $\check{e}_{2i-m_1,2j-m_2}^k = f_{2i-m_1,2j-m_2}^k - (\check{\mathcal{P}}_{k-1}^k f^{k-1})_{2i-m_1,2j-m_2}, \quad 0 \leq m_1, m_2 \leq 1$. A esta estrategia la denotaremos (E1).

La transformada directa sería aplicar el Algoritmo 2.13 pero modificando el predictor por $(\check{\mathcal{P}}_{k-1}^k f^{k-1})$, los errores por $\check{e}^k$ y los detalles por $(\check{d}_1^k, \check{d}_2^k, \check{d}_3^k)$.

No es difícil probar que $\check{d}_l^k = d_l^k$, $l = 1, 2, 3$ con lo que la transformada directa será aplicar exactamente el Algoritmo 2.13.

La transformada inversa si que cambiaría quedando:

Algoritmo 2.15. *Transformación inversa (E1) 2D*

$(\hat{f}^0, \hat{d}_1^1, \hat{d}_2^1, \hat{d}_3^1, \ldots, \hat{d}_1^N, \hat{d}_2^N, \hat{d}_3^N) \to \hat{f}^N = M^{-1}(\hat{f}^0, \hat{d}_1^1, \hat{d}_2^1, \hat{d}_3^1, \ldots, \hat{d}_1^N, \hat{d}_2^N, \hat{d}_3^N)$

for $k = 1, \ldots, N$
  for $i, j = 1, \ldots, J_{k-1}$
    $\hat{\xi}_{i,j}^{k-1} = (\mathcal{D}_k^{k-1}\mathcal{P}_{k-1}^k \hat{f}^{k-1})_{i,j}$
    $(\hat{d}_0^k)_{i,j} = \hat{f}_{i,j}^{k-1} - \hat{\xi}_{i,j}^{k-1}$
    $\hat{f}_{2i-1,2j-1}^k = (\mathcal{P}_{k-1}^k \hat{f}^{k-1})_{2i-1,2j-1} + (\hat{d}_0^k)_{i,j} + (\hat{d}_1^k)_{i,j} + (\hat{d}_2^k)_{i,j} + (\hat{d}_3^k)_{i,j}$
    $\hat{f}_{2i,2j-1}^k = (\mathcal{P}_{k-1}^k \hat{f}^{k-1})_{2i-1,2j-1} + (\hat{d}_0^k)_{i,j} - (\hat{d}_1^k)_{i,j} + (\hat{d}_2^k)_{i,j} - (\hat{d}_3^k)_{i,j}$
    $\hat{f}_{2i-1,2j}^k = (\mathcal{P}_{k-1}^k \hat{f}^{k-1})_{2i-1,2j-1} + (\hat{d}_0^k)_{i,j} + (\hat{d}_1^k)_{i,j} - (\hat{d}_2^k)_{i,j} - (\hat{d}_3^k)_{i,j}$
    $\hat{f}_{2i,2j}^k = (\mathcal{P}_{k-1}^k \hat{f}^{k-1})_{2i,2j} + (\hat{d}_0^k)_{i,j} - (\hat{d}_1^k)_{i,j} - (\hat{d}_2^k)_{i,j} + (\hat{d}_3^k)_{i,j}$
  end
end

Veamos un lema sencillo que nos servirá en la siguiente explicación.

Lema 2.4. *Sea $(V^N, ||\cdot||)$ un espacio de dimensión finita normado y sea $(\mathcal{D}_k^{k-1}, \mathcal{P}_{k-1}^k)_k$ y sea e^k el error definido como $e^k = f^k - \mathcal{P}_{k-1}^k f^{k-1}$ entonces*

$$e^k \in \mathcal{N}(\mathcal{D}_k^{k-1}) \quad \to \mathcal{D}_k^{k-1}\mathcal{P}_{k-1}^k = I_{V^{k-1}}$$

Demostración Si $e^k \in \mathcal{N}(\mathcal{D}_k^{k-1})$ entonces por definición de e^k y por la linealidad del operador $\mathcal{D}_k^{k-1}$ tenemos que:

$$0 = \mathcal{D}_k^{k-1} e^k = \mathcal{D}_k^{k-1}(f^k - \mathcal{P}_{k-1}^k f^{k-1}) = \mathcal{D}_k^{k-1} f^k - \mathcal{D}_k^{k-1}\mathcal{P}_{k-1}^k f^{k-1},$$

así $f^{k-1} = \mathcal{D}_k^{k-1}\mathcal{P}_{k-1}^k f^{k-1}$. El caso inverso está probado en la §2.2 Ec. (2.6). ■

Vamos a presentar ahora una nueva estrategia. Supongamos que el operador predicción no es consistente, i. e. $\mathcal{D}_k^{k-1}\mathcal{P}_{k-1}^k f^{k-1} \neq f^{k-1}$. Calculamos el error:

$$e^k_{2i-m_1,2j-m_2} = f^k_{2i-m_1,2j-m_2} - (\mathcal{P}_{k-1}^k f^{k-1})_{2i-m_1,2j-m_2} \; 0 \leq m_1, m_2 \leq 1, \quad (2.57)$$

entonces definimos

$$(d_0^k)_{i,j} := (\mathcal{D}_k^{k-1} e^k)_{i,j} = \frac{1}{4}(e^k_{2i-1,2j-1} + e^k_{2i,2j-1} + e^k_{2i-1,2j} + e^k_{2i,2j}) = f^{k-1}_{i,j} - \xi^{k-1}_{i,j}, \quad (2.58)$$

con $\xi^{k-1}_{i,j}$ visto en (2.55) dependiente únicamente del nivel $k-1$ (sería 0 si se cumpliese la condición de consistencia por el Lema 2.4). Sea $U_k : V^k \to S^k$:

$$\begin{aligned}
(d_1^k)_{i,j} &:= (U_k e^k)_{1;i,j} = \frac{1}{4}(e^k_{2i-1,2j-1} - e^k_{2i,2j-1} + e^k_{2i-1,2j} - e^k_{2i,2j}),\\
(d_2^k)_{i,j} &:= (U_k e^k)_{2;i,j} = \frac{1}{4}(e^k_{2i-1,2j-1} + e^k_{2i,2j-1} - e^k_{2i-1,2j} - e^k_{2i,2j}),\\
(d_3^k)_{i,j} &:= (U_k e^k)_{3;i,j} = \frac{1}{4}(e^k_{2i-1,2j-1} - e^k_{2i,2j-1} - e^k_{2i-1,2j} + e^k_{2i,2j}),
\end{aligned} \quad (2.59)$$

donde S^k es el espacio que cumple $V^k = V^{k-1} \oplus S^k$.

Es fácil probar que $f^k \equiv (f^{k-1}, d_1^k, d_2^k, d_3^k)$. Una implicación es directa. Supongamos ahora que tenemos $(f^{k-1}, d_1^k, d_2^k, d_3^k)$. A partir de f^{k-1} obtenemos d_0^k. Las ecuaciones (2.59) junto con (2.58) forman un sistema lineal determinado y completo cuya solución es e^k. A partir de e^k y f^{k-1}, utilizando (2.57), obtenemos f^k.

La transformación directa para la estrategia (AY) sería:

Algoritmo 2.16. *Transformación directa (AY) 2D*

$$f^N \to (f^0, d_1^1, d_2^1, d_3^1, \ldots, d_1^N, d_2^N, d_3^N) :$$

for $k = N, \ldots, 1$
  for $i, j = 1, \ldots, J_{k-1}$
    $f_{i,j}^{k-1} = \frac{1}{4}(f_{2i-1,2j-1}^k + f_{2i,2j-1}^k + f_{2i-1,2j}^k + f_{2i,2j}^k)$,
    $e_{2i-1,2j-1}^k = f_{2i-1,2j-1}^k - (\mathcal{P}_{k-1}^k f^{k-1})_{2i-1,2j-1}$,
    $e_{2i,2j-1}^k = f_{2i,2j-1}^k - (\mathcal{P}_{k-1}^k f^{k-1})_{2i,2j-1}$,
    $e_{2i-1,2j}^k = f_{2i-1,2j}^k - (\mathcal{P}_{k-1}^k f^{k-1})_{2i-1,2j}$,
    $e_{2i,2j}^k = f_{2i,2j}^k - (\mathcal{P}_{k-1}^k f^{k-1})_{2i,2j}$,
    $(d_1^k)_{i,j} = \frac{1}{4}(e_{2i-1,2j-1}^k - e_{2i,2j-1}^k + e_{2i-1,2j}^k - e_{2i,2j}^k)$,
    $(d_2^k)_{i,j} = \frac{1}{4}(e_{2i-1,2j-1}^k + e_{2i,2j-1}^k - e_{2i-1,2j}^k - e_{2i,2j}^k)$,
    $(d_3^k)_{i,j} = \frac{1}{4}(e_{2i-1,2j-1}^k - e_{2i,2j-1}^k - e_{2i-1,2j}^k + e_{2i,2j}^k)$,
  end
end

Al igual que en dimensión 1 se puede probar que la transformada directa (AY) 2D (Algoritmo 2.16) es exactamente la misma que la transformada directa (E1) 2D y la (CA) 2D (Algoritmo 2.13). La diferencia entre las estrategias (E1) y (AY) radica en la transformada inversa que para la estrategia (AY) sería:

Algoritmo 2.17. *Transformación inversa (AY) 2D*

$(\hat{f}^0, \hat{d}_1^1, \hat{d}_2^1, \hat{d}_3^1, \ldots, \hat{d}_1^N, \hat{d}_2^N, \hat{d}_3^N) \to \hat{f}^N = M^{-1}(\hat{f}^0, \hat{d}_1^1, \hat{d}_2^1, \hat{d}_3^1, \ldots, \hat{d}_1^N, \hat{d}_2^N, \hat{d}_3^N)$

Sea $\varepsilon = (\varepsilon_k)$ *y* $0 \le \kappa \le 1$
for $k = 1, \ldots, N$
  for $i, j = 1, \ldots, J_{k-1}$
    $\hat{\xi}_{i,j}^{k-1} = (\mathcal{D}_k^{k-1}\mathcal{P}_{k-1}^k \hat{f}^{k-1})_{i,j}$
    $(\tilde{d}_0^k)_{i,j} = \hat{f}_{i,j}^{k-1} - \hat{\xi}_{i,j}^{k-1}$
    $(\hat{d}_0^k)_{i,j} = \mathrm{quad}((\tilde{d}_0^k)_{i,j}, \kappa\varepsilon_k)$
    $\hat{f}_{2i-1,2j-1}^k = (\mathcal{P}_{k-1}^k \hat{f}^{k-1})_{2i-1,2j-1} + (\hat{d}_0^k)_{i,j} + (\hat{d}_1^k)_{i,j} + (\hat{d}_2^k)_{i,j} + (\hat{d}_3^k)_{i,j}$
    $\hat{f}_{2i,2j-1}^k = (\mathcal{P}_{k-1}^k \hat{f}^{k-1})_{2i-1,2j-1} + (\hat{d}_0^k)_{i,j} - (\hat{d}_1^k)_{i,j} + (\hat{d}_2^k)_{i,j} - (\hat{d}_3^k)_{i,j}$
    $\hat{f}_{2i-1,2j}^k = (\mathcal{P}_{k-1}^k \hat{f}^{k-1})_{2i-1,2j-1} + (\hat{d}_0^k)_{i,j} + (\hat{d}_1^k)_{i,j} - (\hat{d}_2^k)_{i,j} - (\hat{d}_3^k)_{i,j}$
    $\hat{f}_{2i,2j}^k = (\mathcal{P}_{k-1}^k \hat{f}^{k-1})_{2i,2j} + (\hat{d}_0^k)_{i,j} - (\hat{d}_1^k)_{i,j} - (\hat{d}_2^k)_{i,j} + (\hat{d}_3^k)_{i,j}$
  end
end

Al igual que en una dimensión si $(\tilde{d}_0^k)$ no es modificado ($\kappa = 0$) las transformaciones inversas son igual para (AY) 2D y (E1) 2D ya que

$$\begin{aligned} f_{2i-1,2j-1}^k &= (\mathcal{P}_{k-1}^k f^{k-1})_{2i-1,2j-1} + (d_0^k)_{i,j} + (d_1^k)_{i,j} + (d_2^k)_{i,j} + (d_3^k)_{i,j} \\ &= \left((\mathcal{P}_{k-1}^k f^{k-1})_{2i-1,2j-1} + f_{i,j}^{k-1} - \xi_{i,j}^{k-1} \right) + (d_1^k)_{i,j} + (d_2^k)_{i,j} + (d_3^k)_{i,j} \\ &= (\check{\mathcal{P}}_{k-1}^k f^{k-1})_{2i-1,2j-1} + (d_1^k)_{i,j} + (d_2^k)_{i,j} + (d_3^k)_{i,j}. \end{aligned}$$

Si el operador es consistente entonces el sumando $f_{i,j}^{k-1} - \xi_{i,j}^{k-1} = 0$ y tendríamos que los Algoritmos 2.15 y 2.17 son equivalentes y que, por tanto, los dos esquemas son iguales.

Veamos cual es la diferencia entre ambas estrategias:

Celda $(2i-1, 2j-1)$:

(E1): $f_{2i-1,2j-1}^k = \underbrace{(\mathcal{P}_{k-1}^k f^{k-1})_{2i-1,2j-1} + f_{i,j}^{k-1} - \xi_{i,j}^{k-1}}_{(\check{\mathcal{P}}_{k-1}^k f^{k-1})_{2i-1,2j-1}} + \underbrace{(d_1^k)_{i,j} + (d_2^k)_{i,j} + (d_3^k)_{i,j}}_{\check{e}_{2i-1,2j-1}^k}$

(AY): $f_{2i-1,2j-1}^k = \underbrace{(\mathcal{P}_{k-1}^k f^{k-1})_{2i-1,2j-1}}_{(\mathcal{P}_{k-1}^k f^{k-1})_{2i-1,2j-1}} + \underbrace{f_{i,j}^{k-1} - \xi_{i,j}^{k-1} + (d_1^k)_{i,j} + (d_2^k)_{i,j} + (d_3^k)_{i,j}}_{e_{2i-1,2j-1}^k}$

Celda $(2i, 2j-1)$:

(E1): $f^k_{2i,2j-1} = \underbrace{(\mathcal{P}^k_{k-1}f^{k-1})_{2i,2j-1} + f^{k-1}_{i,j} - \xi^{k-1}_{i,j}}_{(\check{\mathcal{P}}^k_{k-1}f^{k-1})_{2i,2j-1}} \underbrace{-(d^k_1)_{i,j} + (d^k_2)_{i,j} - (d^k_3)_{i,j}}_{\check{e}^k_{2i,2j-1}}$

(AY): $f^k_{2i,2j-1} = \underbrace{(\mathcal{P}^k_{k-1}f^{k-1})_{2i,2j-1}}_{(\mathcal{P}^k_{k-1}f^{k-1})_{2i,2j-1}} + \underbrace{f^{k-1}_{i,j} - \xi^{k-1}_{i,j} - (d^k_1)_{i,j} + (d^k_2)_{i,j} - (d^k_3)_{i,j}}_{e^k_{2i,2j-1}}$

Celda $(2i-1, 2j)$:

(E1): $f^k_{2i-1,2j} = \underbrace{(\mathcal{P}^k_{k-1}f^{k-1})_{2i-1,2j} + f^{k-1}_{i,j} - \xi^{k-1}_{i,j}}_{(\check{\mathcal{P}}^k_{k-1}f^{k-1})_{2i-1,2j}} + \underbrace{(d^k_1)_{i,j} - (d^k_2)_{i,j} - (d^k_3)_{i,j}}_{\check{e}^k_{2i-1,2j}}$

(AY): $f^k_{2i-1,2j} = \underbrace{(\mathcal{P}^k_{k-1}f^{k-1})_{2i-1,2j}}_{(\mathcal{P}^k_{k-1}f^{k-1})_{2i-1,2j}} + \underbrace{f^{k-1}_{i,j} - \xi^{k-1}_{i,j} + (d^k_1)_{i,j} - (d^k_2)_{i,j} - (d^k_3)_{i,j}}_{e^k_{2i-1,2j}}$

Celda $(2i, 2j)$:

(E1): $f^k_{2i,2j} = \underbrace{(\mathcal{P}^k_{k-1}f^{k-1})_{2i,2j} + f^{k-1}_{i,j} - \xi^{k-1}_{i,j}}_{(\check{\mathcal{P}}^k_{k-1}f^{k-1})_{2i,2j}} \underbrace{-(d^k_1)_{i,j} - (d^k_2)_{i,j} + (d^k_3)_{i,j}}_{\check{e}^k_{2i,2j}}$

(AY): $f^k_{2i,2j} = \underbrace{(\mathcal{P}^k_{k-1}f^{k-1})_{2i,2j}}_{(\mathcal{P}^k_{k-1}f^{k-1})_{2i,2j}} + \underbrace{f^{k-1}_{i,j} - \xi^{k-1}_{i,j} - (d^k_1)_{i,j} - (d^k_2)_{i,j} + (d^k_3)_{i,j}}_{e^k_{2i,2j}}$

En la estrategia (E1) se modifica el operador predicción para conseguir la consistencia introduciendo un sumando poco natural que no surge de ninguno de los métodos propuestos (interpolación, aprendizaje o aproximación por núcleos). Sin embargo en la estrategia (AY) no modificamos el operador predicción sino los errores de tal manera que si los cuantizamos seguimos conservando la naturaleza del método utilizado y si no cuantizamos tendremos los mismos resultados que si utilizamos un operador consistente.

Hemos visto que el orden se conservaba utilizando la estrategia (AY), veamos ahora la estabilidad.

Recordamos la notación:

$$\begin{aligned} f^k_{2i-1,2j-1} &= (\mathcal{P}^k_{k-1}f^{k-1})_{2i-1,2j-1} + (d^k_0)_{i,j} + (d^k_1)_{i,j} + (d^k_2)_{i,j} + (d^k_3)_{i,j}, \\ f^k_{2i,2j-1} &= (\mathcal{P}^k_{k-1}f^{k-1})_{2i,2j-1} + (d^k_0)_{i,j} - (d^k_1)_{i,j} + (d^k_2)_{i,j} - (d^k_3)_{i,j}, \\ f^k_{2i-1,2j} &= (\mathcal{P}^k_{k-1}f^{k-1})_{2i-1,2j} + (d^k_0)_{i,j} + (d^k_1)_{i,j} - (d^k_2)_{i,j} - (d^k_3)_{i,j}, \\ f^k_{2i,2j} &= (\mathcal{P}^k_{k-1}f^{k-1})_{2i,2j} + (d^k_0)_{i,j} - (d^k_1)_{i,j} - (d^k_2)_{i,j} + (d^k_3)_{i,j}, \end{aligned} \tag{2.60}$$

con

$$d^k_0 = f^{k-1} - \mathcal{D}^{k-1}_k \mathcal{P}^k_{k-1} f^{k-1}, \tag{2.61}$$

Llamaremos:

$$\begin{aligned} r^k_{2i-1,2j-1} =& (d^k_0)_{i,j} + (d^k_1)_{i,j} + (d^k_2)_{i,j} + (d^k_3)_{i,j}, \\ r^k_{2i,2j-1} =& (d^k_0)_{i,j} - (d^k_1)_{i,j} + (d^k_2)_{i,j} - (d^k_3)_{i,j}, \\ r^k_{2i-1,2j} =& (d^k_0)_{i,j} + (d^k_1)_{i,j} - (d^k_2)_{i,j} - (d^k_3)_{i,j}, \\ r^k_{2i,2j} =& (d^k_0)_{i,j} - (d^k_1)_{i,j} - (d^k_2)_{i,j} + (d^k_3)_{i,j}. \end{aligned} \tag{2.62}$$

$$\begin{aligned} \hat{f}^k_{2i-1,2j-1} &= (\mathcal{P}^k_{k-1}\hat{f}^{k-1})_{2i-1,2j-1} + (\hat{d}^k_0)_{i,j} + (\hat{d}^k_1)_{i,j} + (\hat{d}^k_2)_{i,j} + (\hat{d}^k_3)_{i,j}, \\ \hat{f}^k_{2i,2j-1} &= (\mathcal{P}^k_{k-1}\hat{f}^{k-1})_{2i,2j-1} + (\hat{d}^k_0)_{i,j} - (\hat{d}^k_1)_{i,j} + (\hat{d}^k_2)_{i,j} - (\hat{d}^k_3)_{i,j}, \\ \hat{f}^k_{2i-1,2j} &= (\mathcal{P}^k_{k-1}\hat{f}^{k-1})_{2i-1,2j} + (\hat{d}^k_0)_{i,j} + (\hat{d}^k_1)_{i,j} - (\hat{d}^k_2)_{i,j} - (\hat{d}^k_3)_{i,j}, \\ \hat{f}^k_{2i,2j} &= (\mathcal{P}^k_{k-1}\hat{f}^{k-1})_{2i,2j} + (\hat{d}^k_0)_{i,j} - (\hat{d}^k_1)_{i,j} - (\hat{d}^k_2)_{i,j} + (\hat{d}^k_3)_{i,j}, \end{aligned} \tag{2.63}$$

con

$$\hat{d}^k_l = \mathit{quad}(d^k_l, \varepsilon), \quad l = 1, 2, 3. \tag{2.64}$$

$$\hat{d}^k_0 = \mathit{quad}(\tilde{d}^k_0, \kappa\varepsilon), \text{ con } \tilde{d}^k_0 := \hat{f}^{k-1} - \mathcal{D}^{k-1}_k \mathcal{P}^k_{k-1}\hat{f}^{k-1} \tag{2.65}$$

Llamaremos

$$\begin{aligned} \hat{r}^k_{2i-1,2j-1} =& (\hat{d}^k_0)_{i,j} + (\hat{d}^k_1)_{i,j} + (\hat{d}^k_2)_{i,j} + (\hat{d}^k_3)_{i,j}, \\ \hat{r}^k_{2i,2j-1} =& (\hat{d}^k_0)_{i,j} - (\hat{d}^k_1)_{i,j} + (\hat{d}^k_2)_{i,j} - (\hat{d}^k_3)_{i,j}, \\ \hat{r}^k_{2i-1,2j} =& (\hat{d}^k_0)_{i,j} + (\hat{d}^k_1)_{i,j} - (\hat{d}^k_2)_{i,j} - (\hat{d}^k_3)_{i,j}, \\ \hat{r}^k_{2i,2j} =& (\hat{d}^k_0)_{i,j} - (\hat{d}^k_1)_{i,j} - (\hat{d}^k_2)_{i,j} + (\hat{d}^k_3)_{i,j}. \end{aligned} \tag{2.66}$$

Lema 2.5. *Sean $a, b, c, d \in \mathbb{R}$ entonces se cumple que:*

$$|a+b+c+d| + |a-b+c-d| + |a+b-c-d| + |a-b-c+d| \leq 4(|a| + |b| + |c| + |d|) \tag{2.67}$$

$$\max\{|a+b+c+d|, |a-b+c-d|, |a+b-c-d|, |a-b-c+d|\} \leq 4\max\{|a|, |b|, |c|, |d|\} \tag{2.68}$$

$$(a+b+c+d)^2 + (a-b+c-d)^2 + (a+b-c-d)^2 + (a-b-c+d)^2 = 4(a^2+b^2+c^2+d^2) \tag{2.69}$$

Demostración La demostración de la Ec. (2.67) se da por las relaciones:

$$\begin{aligned}
|a+b+c+d| &\leq |a|+|b|+|c|+|d|,\\
|a-b+c-d| &\leq |a|+|b|+|c|+|d|,\\
|a+b-c-d| &\leq |a|+|b|+|c|+|d|,\\
|a-b-c+d| &\leq |a|+|b|+|c|+|d|.
\end{aligned}$$

Por las relaciones:

$$\begin{aligned}
\text{máx}\{|a+b+c+d|\} &\leq \text{máx}\{|a|+|b|+|c|+|d|\} \leq 4\,\text{máx}\{|a|,|b|,|c|,|d|\},\\
\text{máx}\{|a-b+c-d|\} &\leq \text{máx}\{|a|+|b|+|c|+|d|\} \leq 4\,\text{máx}\{|a|,|b|,|c|,|d|\},\\
\text{máx}\{|a+b-c-d|\} &\leq \text{máx}\{|a|+|b|+|c|+|d|\} \leq 4\,\text{máx}\{|a|,|b|,|c|,|d|\},\\
\text{máx}\{|a-b-c+d|\} &\leq \text{máx}\{|a|+|b|+|c|+|d|\} \leq 4\,\text{máx}\{|a|,|b|,|c|,|d|\},
\end{aligned}$$

obtenemos la Ec. (2.68)

Y, por último, la Ec. (2.69) la podemos ver teniendo en cuenta que:

$$\begin{aligned}
(a+b+c+d)^2 &= a^2+b^2+c^2+d^2\\
&= 2ab+2ac+2ad+2bc+2bd+2cd,\\
(a-b+c-d)^2 &= a^2+b^2+c^2+d^2\\
&= -2ab+2ac-2ad-2bc+2bd-2cd,\\
(a+b-c-d)^2 &= a^2+b^2+c^2+d^2\\
&= 2ab-2ac-2ad-2bc-2bd+2cd,\\
(a-b-c+d)^2 &= a^2+b^2+c^2+d^2\\
&= -2ab-2ac+2ad+2bc-2bd-2cd.
\end{aligned}$$

■

Proposición 2.2. *Con la notación utilizada durante todo la sección tenemos que:*

$$||r^k - \hat{r}^k||_\infty \leq 4||||d^k - \hat{d}^k||||_\infty,$$
$$||r^k - \hat{r}^k||_1 \leq ||||d^k - \hat{d}^k||||_1,$$
$$||r^k - \hat{r}^k||_2^2 = ||||d^k - \hat{d}^k||||_2^2,$$

donde:

$$||||d^k - \hat{d}^k||||_2^2 = ||d_0^k - \hat{d}_0^k||_2^2 + ||d_1^k - \hat{d}_1^k||_2^2 + ||d_2^k - \hat{d}_2^k||_2^2 + ||d_3^k - \hat{d}_3^k||_2^2$$
$$||||d^k - \hat{d}^k||||_1 = ||d_0^k - \hat{d}_0^k||_1 + ||d_1^k - \hat{d}_1^k||_1 + ||d_2^k - \hat{d}_2^k||_1 + ||d_3^k - \hat{d}_3^k||_1$$
$$||||d^k - \hat{d}^k||||_\infty = \text{máx}\{||d_0^k - \hat{d}_0^k||_\infty, ||d_1^k - \hat{d}_1^k||_\infty, ||d_2^k - \hat{d}_2^k||_\infty, ||d_3^k - \hat{d}_3^k||_\infty\}.$$

Demostración Para demostrar la proposición denotaremos (eliminando el nivel k por comodidad) como:

$$a_{i,j} = (d_0^k)_{i,j} - (\hat{d}_0^k)_{i,j}, \qquad b_{i,j} = (d_1^k)_{i,j} - (\hat{d}_1^k)_{i,j},$$
$$c_{i,j} = (d_2^k)_{i,j} - (\hat{d}_2^k)_{i,j}, \qquad d_{i,j} = (d_3^k)_{i,j} - (\hat{d}_3^k)_{i,j}.$$

Con esta notación tenemos que:

$$|r_{2i-1,2j-1}^k - \hat{r}_{2i-1,2j-1}^k| = |a_{i,j} + b_{i,j} + c_{i,j} + d_{i,j}|$$
$$|r_{2i,2j-1}^k - \hat{r}_{2i,2j-1}^k| = |a_{i,j} - b_{i,j} + c_{i,j} - d_{i,j}|$$
$$|r_{2i-1,2j}^k - \hat{r}_{2i-1,2j}^k| = |a_{i,j} + b_{i,j} - c_{i,j} - d_{i,j}|$$
$$|r_{2i,2j}^k - \hat{r}_{2i,2j}^k| = |a_{i,j} - b_{i,j} - c_{i,j} + d_{i,j}|.$$

Así pues

$$\begin{aligned}||r^k - \hat{r}^k||_\infty &= \max_{i,j=1,\ldots,J_{k-1}} \{|r_{2i-1,2j-1}^k - \hat{r}_{2i-1,2j-1}^k|, |r_{2i,2j-1}^k - \hat{r}_{2i,2j-1}^k|, \\ &\qquad |r_{2i-1,2j}^k - \hat{r}_{2i-1,2j}^k|, |r_{2i,2j}^k - \hat{r}_{2i,2j}^k|\} \\ &= \max_{i,j=1,\ldots,J_{k-1}} \{|a_{i,j} + b_{i,j} + c_{i,j} + d_{i,j}|, |a_{i,j} - b_{i,j} + c_{i,j} - d_{i,j}|, \\ &\qquad |a_{i,j} + b_{i,j} - c_{i,j} - d_{i,j}|, |a_{i,j} - b_{i,j} - c_{i,j} + d_{i,j}|\} \\ &\leq 4 \max_{i,j=1,\ldots,J_{k-1}} \{|a_{i,j}|, |b_{i,j}|, |c_{i,j}|, |d_{i,j}|\} = 4||||d^k - \hat{d}^k||||_\infty.\end{aligned}$$

Con la notación utilizada y el lema anterior probamos que:

$$\begin{aligned}
J_k^2||r^k-\hat{r}^k||_1 &= \sum_{i,j=1}^{J_{k-1}} |r^k_{2i-1,2j-1}-\hat{r}^k_{2i-1,2j-1}| + |r^k_{2i,2j-1}-\hat{r}^k_{2i,2j-1}| + \\
&\quad + |r^k_{2i-1,2j}-\hat{r}^k_{2i-1,2j}| + |r^k_{2i,2j}-\hat{r}^k_{2i,2j}| \\
&= \sum_{i,j=1}^{J_{k-1}} |a_{i,j}+b_{i,j}+c_{i,j}+d_{i,j}| + |a_{i,j}-b_{i,j}+c_{i,j}-d_{i,j}| + \\
&\quad + |a_{i,j}+b_{i,j}-c_{i,j}-d_{i,j}| + |a_{i,j}-b_{i,j}-c_{i,j}+d_{i,j}| \\
&\leq 4\sum_{i,j=1}^{J_{k-1}} |a_{i,j}| + |b_{i,j}| + |c_{i,j}| + |d_{i,j}| = 4J_{k-1}^2||||d^k-\hat{d}^k||||_1.
\end{aligned}$$

Análogamente utilizando el Lema 2.5 para norma 2

$$\begin{aligned}
J_k^2||r^k-\hat{r}^k||_2^2 &= \sum_{i,j=1}^{J_{k-1}} |r^k_{2i-1,2j-1}-\hat{r}^k_{2i-1,2j-1}|^2 + |r^k_{2i,2j-1}-\hat{r}^k_{2i,2j-1}|^2 + \\
&\quad + |r^k_{2i-1,2j}-\hat{r}^k_{2i-1,2j}|^2 + |r^k_{2i,2j}-\hat{r}^k_{2i,2j}|^2 \\
&= \sum_{i,j=1}^{J_{k-1}} |a_{i,j}+b_{i,j}+c_{i,j}+d_{i,j}|^2 + |a_{i,j}-b_{i,j}+c_{i,j}-d_{i,j}|^2 + \\
&\quad + |a_{i,j}+b_{i,j}-c_{i,j}-d_{i,j}|^2 + |a_{i,j}-b_{i,j}-c_{i,j}+d_{i,j}|^2 \\
&= 4\sum_{i,j=1}^{J_{k-1}} |a_{i,j}|^2 + |b_{i,j}|^2 + |c_{i,j}|^2 + |d_{i,j}|^2 = 4J_{k-1}^2||||d^k-\hat{d}^k||||_2^2.
\end{aligned}$$

■

Lema 2.6. *Siguiendo la notación de todo el capítulo, si $\mathcal{P}_{k-1}^k$ es lineal tenemos que:*

$$||d_0^k-\hat{d}_0^k||_\alpha \leq ||\tilde{d}_0^k-\hat{d}_0^k||_\alpha + ||I_{V^{k-1}}-\mathcal{D}_k^{k-1}\mathcal{P}_{k-1}^k||_\alpha||f^{k-1}-\hat{f}^{k-1}||_\alpha \quad (2.70)$$

donde

$$\hat{d}_0^k = \text{quad}(\tilde{d}_0^k, \kappa\varepsilon), \text{ con } \quad \tilde{d}_0^k = \hat{f}^{k-1}-\mathcal{D}_k^{k-1}\mathcal{P}_{k-1}^k\hat{f}^{k-1} \text{ y } \alpha = 1, 2, \infty.$$

Demostración

$$\begin{aligned}||d_0^k - \hat{d}_0^k||_\alpha &= ||d_0^k - \tilde{d}_0^k + \tilde{d}_0^k - \hat{d}_0^k||_\alpha \\ &\leq ||\tilde{d}_0^k - \hat{d}_0^k||_\alpha + ||f^{k-1} - \mathcal{D}_k^{k-1}\mathcal{P}_{k-1}^k f^{k-1} - \hat{f}^{k-1} + \mathcal{D}_k^{k-1}\mathcal{P}_{k-1}^k \hat{f}^{k-1}||_\alpha \\ &= ||\tilde{d}_0^k - \hat{d}_0^k||_\alpha + ||(I_{V^{k-1}} - \mathcal{D}_k^{k-1}\mathcal{P}_{k-1}^k)(f^{k-1} - \hat{f}^{k-1})||_\alpha \\ &\leq ||\tilde{d}_0^k - \hat{d}_0^k||_\alpha + ||I_{V^{k-1}} - \mathcal{D}_k^{k-1}\mathcal{P}_{k-1}^k||_\alpha ||f^{k-1} - \hat{f}^{k-1}||_\alpha.\end{aligned} \tag{2.71}$$

■

Nota 2.2. *Si* $\alpha = 2$ *entonces:*

$$\begin{aligned}||d_0^k - \hat{d}_0^k||_2^2 &\leq (||\tilde{d}_0^k - \hat{d}_0^k||_2 + ||I_{V^{k-1}} - \mathcal{D}_k^{k-1}\mathcal{P}_{k-1}^k||_2 ||f^{k-1} - \hat{f}^{k-1}||_2)^2 \\ &\leq 2(||\tilde{d}_0^k - \hat{d}_0^k||_2^2 + ||I_{V^{k-1}} - \mathcal{D}_k^{k-1}\mathcal{P}_{k-1}^k||_2^2 ||f^{k-1} - \hat{f}^{k-1}||_2^2).\end{aligned} \tag{2.72}$$

Corolario 2.3. *Con la notación utilizada durante toda la sección. Sea un esquema de multiresolución dos dimensional* $(\mathcal{D}_k^{k-1}, \mathcal{P}_{k-1}^k)_k$ *no necesariamente consistente en el contexto de medias en celda, siendo el operador predicción lineal, es decir:*

$$||\mathcal{P}_{k-1}^k f^{k-1} - \mathcal{P}_{k-1}^k \hat{f}^{k-1}||_\alpha \leq ||\mathcal{P}_{k-1}^k||_\alpha ||f^{k-1} - \hat{f}^{k-1}||_\alpha, \quad \forall\, f^{k-1}, \hat{f}^{k-1} \in V^{k-1}, \tag{2.73}$$

siendo $\alpha = 1, 2, \infty$*, entonces si utilizamos los algoritmos de descomposición y composición siguiendo la estrategia (AY) tenemos que:*
Si $\alpha = 1, 2$ *entonces:*

$$\begin{aligned}||f^L - \hat{f}^L||_\alpha^\alpha \leq & K_\alpha^L ||f^0 - \hat{f}^0||_\alpha^\alpha + \\ & + \sum_{k=0}^{L-1} K_\alpha^k \big(4^{\alpha-1}||\tilde{d}_0^{L-k} - \hat{d}_0^{L-k}||_\alpha^\alpha + 2^{\alpha-1}\big(||d_1^{L-k} - \hat{d}_1^{L-k}||_\alpha^\alpha + \\ & \quad + ||d_2^{L-k} - \hat{d}_2^{L-k}||_\alpha^\alpha + ||d_3^{L-k} - \hat{d}_3^{L-k}||_\alpha^\alpha\big)\big)\end{aligned}$$

con $K_\alpha = \max_{k=1,\dots,L}(2^{\alpha-1}||\mathcal{P}_{k-1}^k||_\alpha^\alpha + 4^{\alpha-1}||I_{V^{k-1}} - \mathcal{D}_k^{k-1}\mathcal{P}_{k-1}^k||_\alpha^\alpha)$.
Si $\alpha = \infty$

$$\begin{aligned}||f^L - \hat{f}^L||_\infty \leq & K_\infty^L \max_{k=1,\dots,L}\{||f^0 - \hat{f}^0||_\infty, ||\tilde{d}_0^k - \hat{d}_0^k||_\infty, ||d_1^k - \hat{d}_1^k||_\infty, \\ & ||d_2^k - \hat{d}_2^k||_\infty, ||d_3^k - \hat{d}_3^k||_\infty\}\end{aligned}$$

siendo $K_\infty = \max_{k=1,\dots,L}(||\mathcal{P}_{k-1}^k||_\infty + 4||I_{V^{k-1}} - \mathcal{D}_k^{k-1}\mathcal{P}_{k-1}^k||_\infty + 4)$.

Demostración Tenemos por la hipótesis del teorema y la Proposición 2.2 que:

Si $\alpha = 1, 2$, sea $K_\alpha = \max_{k=1,\dots,L}(2^{\alpha-1}||\mathcal{P}_{k-1}^k||_\alpha^\alpha + 4^{\alpha-1}||I_{V^{k-1}} - \mathcal{D}_k^{k-1}\mathcal{P}_{k-1}^k||_\alpha^\alpha)$ entonces:

$$\begin{aligned}
||f^k - \hat{f}^k||_\alpha^\alpha =& ||\mathcal{P}_{k-1}^k f^{k-1} + r^k - \mathcal{P}_{k-1}^k \hat{f}^{k-1} - \hat{r}^k||_\alpha^\alpha \\
\leq& 2^{\alpha-1}||\mathcal{P}_{k-1}^k||_\alpha^\alpha ||f^{k-1} - \hat{f}^{k-1}||_\alpha^\alpha + 2^{\alpha-1}||r^k - \hat{r}^k||_\alpha^\alpha \\
\leq& 2^{\alpha-1}||\mathcal{P}_{k-1}^k||_\alpha^\alpha ||f^{k-1} - \hat{f}^{k-1}||_\alpha^\alpha + 2^{\alpha-1}||||d^k - \hat{d}^k||||_\alpha^\alpha \\
=& 2^{\alpha-1}||\mathcal{P}_{k-1}^k||_\alpha^\alpha ||f^{k-1} - \hat{f}^{k-1}||_\alpha^\alpha + 2^{\alpha-1}||d_0^k - \hat{d}_0^k||_\alpha^\alpha + \\
& + 2^{\alpha-1}(||d_1^k - \hat{d}_1^k||_\alpha^\alpha + ||d_2^k - \hat{d}_2^k||_\alpha^\alpha + ||d_3^k - \hat{d}_3^k||_\alpha^\alpha) \\
\leq& 2^{\alpha-1}||\mathcal{P}_{k-1}^k||_\alpha^\alpha ||f^{k-1} - \hat{f}^{k-1}||_\alpha^\alpha + \\
& + 4^{\alpha-1}||I_{V^{k-1}} - \mathcal{D}_k^{k-1}\mathcal{P}_{k-1}^k||_\alpha^\alpha ||f^{k-1} - \hat{f}^{k-1}||_\alpha^\alpha + \\
& + 4^{\alpha-1}||\tilde{d}_0^k - \hat{d}_0^k||_\alpha^\alpha + 2^{\alpha-1}(||d_1^k - \hat{d}_1^k||_\alpha^\alpha + ||d_2^k - \hat{d}_2^k||_\alpha^\alpha + ||d_3^k - \hat{d}_3^k||_\alpha^\alpha) \\
=& K_\alpha ||f^{k-1} - \hat{f}^{k-1}||_\alpha^\alpha + 4^{\alpha-1}||\tilde{d}_0^k - \hat{d}_0^k||_\alpha^\alpha + 2^{\alpha-1}||d_1^k - \hat{d}_1^k||_\alpha^\alpha + \\
& + 2^{\alpha-1}||d_2^k - \hat{d}_2^k||_\alpha^\alpha + 2^{\alpha-1}||d_3^k - \hat{d}_3^k||_\alpha^\alpha.
\end{aligned}$$

Si $\alpha = \infty$, sea $K_\infty = \max_{k=1,\dots,L}(||\mathcal{P}_{k-1}^k||_\infty + 4||I_{V^{k-1}} - \mathcal{D}_k^{k-1}\mathcal{P}_{k-1}^k||_\infty + 4)$ tenemos que:

$$\begin{aligned}
||f^k - \hat{f}^k||_\infty =& ||\mathcal{P}_{k-1}^k f^{k-1} + r^k - \mathcal{P}_{k-1}^k \hat{f}^{k-1} - \hat{r}^k||_\infty \\
\leq& ||\mathcal{P}_{k-1}^k||_\infty ||f^{k-1} - \hat{f}^{k-1}||_\infty + ||r^k - \hat{r}^k||_\infty \\
\leq& ||\mathcal{P}_{k-1}^k||_\infty ||f^{k-1} - \hat{f}^{k-1}||_\infty + 4||||d^k - \hat{d}^k||||_\infty \\
=& ||\mathcal{P}_{k-1}^k||_\infty ||f^{k-1} - \hat{f}^{k-1}||_\infty + 4\max\{||d_0^k - \hat{d}_0^k||_\infty, \\
& ||d_1^k - \hat{d}_1^k||_\infty, ||d_2^k - \hat{d}_2^k||_\infty, ||d_3^k - \hat{d}_3^k||_\infty\} \\
\leq& ||\mathcal{P}_{k-1}^k||_\infty ||f^{k-1} - \hat{f}^{k-1}||_\infty + 4\max\{||d_0^k - \tilde{d}_0^k||_\infty + \\
& + ||\tilde{d}_0^k - \hat{d}_0^k||_\infty, ||d_1^k - \hat{d}_1^k||_\infty, ||d_2^k - \hat{d}_2^k||_\infty, ||d_3^k - \hat{d}_3^k||_\infty\} \\
\leq& ||\mathcal{P}_{k-1}^k||_\infty ||f^{k-1} - \hat{f}^{k-1}||_\infty + 4\max\{||I_{V^{k-1}} - \mathcal{D}_k^{k-1}\mathcal{P}_{k-1}^k||_\infty ||f^{k-1} - \hat{f}^{k-1}||_\infty \\
& + ||\tilde{d}_0^k - \hat{d}_0^k||_\infty, ||d_1^k - \hat{d}_1^k||_\infty, ||d_2^k - \hat{d}_2^k||_\infty, ||d_3^k - \hat{d}_3^k||_\infty\} \\
=& K_\infty \max\{||f^{k-1} - \hat{f}^{k-1}||_\infty, ||\tilde{d}_0^k - \hat{d}_0^k||_\infty, ||d_1^k - \hat{d}_1^k||_\infty, \\
& ||d_2^k - \hat{d}_2^k||_\infty, ||d_3^k - \hat{d}_3^k||_\infty\}.
\end{aligned}$$

■

Corolario 2.4. *Con la notación utilizada durante toda la sección. Sea un esquema de multiresolución dos dimensional* $(\mathcal{D}_k^{k-1}, \mathcal{P}_{k-1}^k)_k$ *no necesariamente consistente en el contexto de medias en celda, siendo el operador predicción lineal. Si existe* $C_\alpha > 0$ *con* $\alpha = 1, 2, \infty$ *tal que*

$$||\hat{d}_0^k - \tilde{d}_0^k||_\alpha \leq C_\alpha(||f^{k-1} - \hat{f}^{k-1}||_\alpha + ||||d^k - \hat{d}^k||||_\alpha), \ \forall\, k, \tag{2.74}$$

siendo $\alpha = 1, 2$, *y*

$$||\hat{d}_0^k - \tilde{d}_0^k||_\infty \leq C_\infty \,\text{máx}\{||f^{k-1} - \hat{f}^{k-1}||_\alpha, ||||d^k - \hat{d}^k||||_\infty\}, \ \forall\, k; \tag{2.75}$$

entonces los algoritmos de descomposición y composición siguiendo la estrategia (AY) son estables.

Demostración La demostración es inmediata utilizando el Corolario 2.3. ■

El Corolario 2.4 se cumple siempre en los casos prácticos.

Veamos, al igual que antes, un ejemplo con el propósito de aclarar las diferencias entre las distintas estrategias.

Hallamos un operador predictor utilizando la extensión tensorial del operador predicción 1D presentado en (2.54) (Fig. 2.19):

$$\begin{cases} (\mathcal{P}_{k-1}^k f^{k-1})_{2i-1,2j-1} = \frac{1}{16} f_{i-1,j-1}^{k-1} + \frac{3}{16} f_{i,j-1}^{k-1} + \frac{3}{16} f_{i-1,j}^{k-1} + \frac{9}{16} f_{i,j}^{k-1}, \\ (\mathcal{P}_{k-1}^k f^{k-1})_{2i,2j-1} = \frac{1}{16} f_{i+1,j-1}^{k-1} + \frac{3}{16} f_{i,j-1}^{k-1} + \frac{3}{16} f_{i+1,j}^{k-1} + \frac{9}{16} f_{i,j}^{k-1}, \\ (\mathcal{P}_{k-1}^k f^{k-1})_{2i-1,2j} = \frac{1}{16} f_{i-1,j+1}^{k-1} + \frac{3}{16} f_{i-1,j}^{k-1} + \frac{3}{16} f_{i,j+1}^{k-1} + \frac{9}{16} f_{i,j}^{k-1}, \\ (\mathcal{P}_{k-1}^k f^{k-1})_{2i,2j} = \frac{1}{16} f_{i+1,j+1}^{k-1} + \frac{3}{16} f_{i,j+1}^{k-1} + \frac{3}{16} f_{i+1,j}^{k-1} + \frac{9}{16} f_{i,j}^{k-1}. \end{cases} \tag{2.76}$$

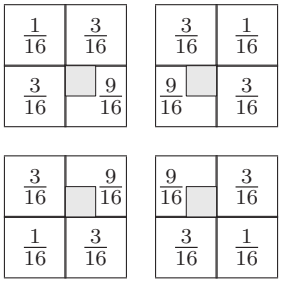


Figura 2.19: *Operador predictor utilizando la estrategia (AY).*

Si utilizamos la estrategia (E1) calcularíamos:

$$f_{i,j}^{k-1} - \xi_{i,j}^{k-1} = -\frac{1}{64}\left(f_{i-1,j-1}^{k-1} + f_{i+1,j-1}^{k-1} + f_{i-1,j+1}^{k-1} + f_{i+1,j+1}^{k-1}\right) - \frac{3}{32}\left(f_{i,j-1}^{k-1} + f_{i-1,j}^{k-1} + f_{i,j+1}^{k-1} + f_{i+1,j}^{k-1}\right) + \frac{7}{16} f_{i,j}^{k-1},$$

y obtendríamos el operador predicción mostrado en la Figura 2.20.

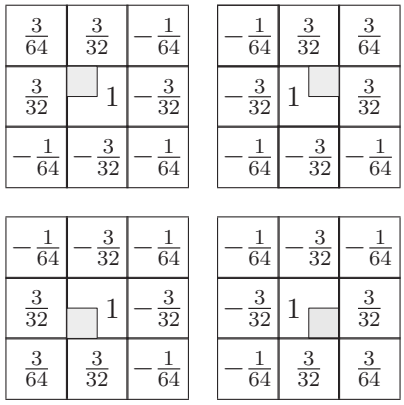


Figura 2.20: *Operador predictor utilizando la estrategia (E1).*

Los métodos (E0) y (E1) producen peores resultados debido a que los operadores aparecen por la construcción de consistencia y no por la naturaleza del operador predicción. Para comprobar esta afirmación vamos a realizar dos experimentos con las imágenes: *lena* y *peppers*.

A estas imágenes les aplicamos las transformaciones directas para las distintas estrategias. Tomaremos $N = 4$.

Perturbaremos los detalles $(d_1^k, d_2^k, d_3^k)_{k=1}^N$ cuantizándolos por medio de la función:

$$(\hat{d}_l^k)_{i,j} = \text{quad}((d_l^k)_{i,j}, \varepsilon_k) = 2\varepsilon_k \left[\frac{(d_l^k)_{i,j}}{2\varepsilon_k}\right], \quad l = 1, 2, 3$$

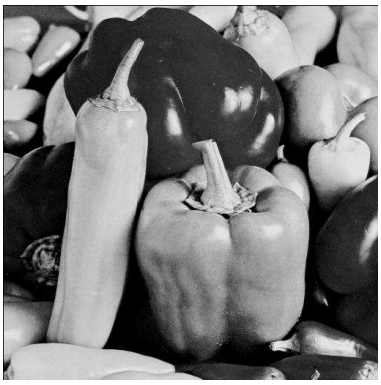
con

$$\varepsilon_{k-1} = \begin{cases} \frac{\varepsilon_k}{2}, & \textbf{si} \quad \varepsilon_k \geq 1/2; \\ 1/2, & \textbf{si} \quad \varepsilon_k < 1/2 \end{cases} \tag{2.77}$$

siendo ε_N dado y $[\cdot]$ el entero obtenido por redondeo.

Al aplicar la transformada inversa para la estrategia (AY) 2D (Algoritmo 2.17) modificaremos los valores d_0^k truncando:

lena



peppers

$$(\hat{d}_0^k)_{i,j} = \begin{cases} (\tilde{d}_0^k)_{i,j}, & \text{si } |(\tilde{d}_0^k)_{i,j}| \geq \kappa\varepsilon_k; \\ 0, & \text{si } |(\tilde{d}_0^k)_{i,j}| < \kappa\varepsilon_k; \end{cases}$$

con $\kappa = \frac{1}{2}$.

Mediremos la capacidad de compresión contando el número de elementos no nulos que quedan después de cuantizar. Los denotaremos como NNZ. La calidad de la imagen reconstruida, $\hat{f}^N$, la mediremos utilizando la norma 2 discreta y el valor PSNR que definimos como:

$$E_2 = \frac{1}{J_N}\left(\sum_{i,j} |f_{i,j}^N - \hat{f}_{i,j}^N|^2\right)^{\frac{1}{2}}, \quad \text{PSNR} = 20\log_{10}\left(\frac{256}{E_2}\right). \tag{2.78}$$

	$\varepsilon_N = 4$		$\varepsilon_N = 16$		$\varepsilon_N = 32$		$\varepsilon_N = 64$	
	NNZ	PSNR	NNZ	PSNR	NNZ	PSNR	NNZ	PSNR
CA_3	35378	37,52	7095	31,22	2719	28,28	883	25,77
(E0)	86132	37,45	21860	28,92	8883	24,99	3378	21,57
(E1)	36750	37,50	7340	31,14	2802	28,23	899	25,74
(AY)	36750	37,51	7340	31,27	2802	28,45	899	25,94

Tabla 2.3: *Resultados obtenidos para la imagen* lena *utilizando CA_3 y las distintas estrategias (E0), (E1) y (AY) con* $\varepsilon_N = 4, 16, 32, 64$.

La diferencia de la estrategia (AY) con el resto de métodos consiste en que la transformada inversa trunca los valores (d_0^k). Esta estrategia conserva el grado de aproximación y el control del error sin perder la naturaleza del operador predictor hallado.

	$\varepsilon_N = 4$		$\varepsilon_N = 16$		$\varepsilon_N = 32$		$\varepsilon_N = 64$	
	NNZ	PSNR	NNZ	PSNR	NNZ	PSNR	NNZ	PSNR
CA_3	40798	37,69	8864	30,85	4138	27,77	1522	24,31
(E0)	89404	37,61	25457	29,01	12574	24,89	5285	20,68
(E1)	41612	37,67	9111	30,80	4221	27,71	1547	24,28
(AY)	41612	37,73	9111	31,00	4221	28,01	1547	24,51

Tabla 2.4: *Resultados obtenidos para la imagen* peppers *utilizando CA_3 y las distintas estrategias (E0), (E1) y (AY) con $\varepsilon_N = 4, 16, 32, 64$.*

Como se puede ver en la Figura 2.21 y en las Tablas 2.3 y 2.4 la estrategia (AY) proporciona similares resultados. Además el caso particular del operador predictor planteado (similar a un esquema B-*spline* en subdivisión, ver p.ej. [33] para más detalle) proporciona una mayor calidad visual (Figuras 2.22 y 2.24) y evita el fenómeno de *gibbs*. Ampliamos una zona de las imágenes (Figuras 2.23 y 2.25) para ver esto con mayor claridad.

Por tanto (AY) nos ofrece una alternativa ante posibles operadores no consistentes abriendo grandes posibilidades para poder introducir predictores no lineales (véase medias PPH o ENO) sin perder la naturaleza de procedencia de éstos.

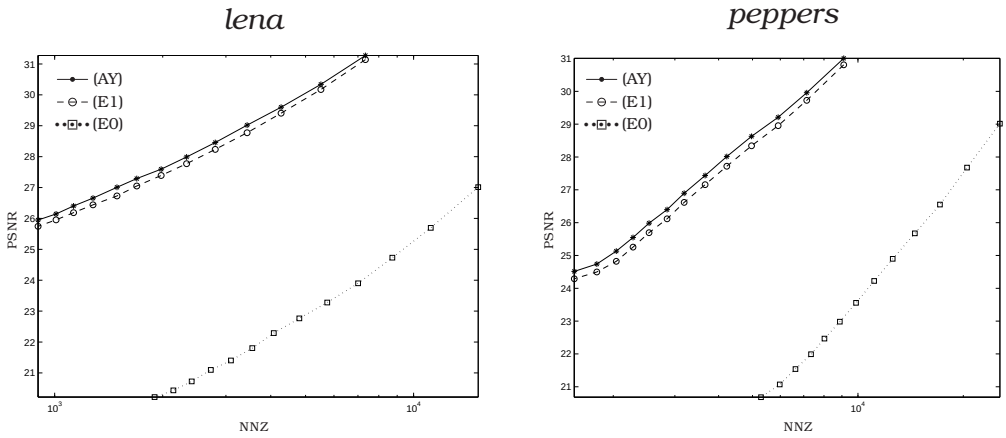


Figura 2.21: *Curva* PSNR *y elementos no nulos (*NNZ*) utilizando las estrategias (E0), (E1) y (AY) para las imágenes* lena *y* peppers*.*

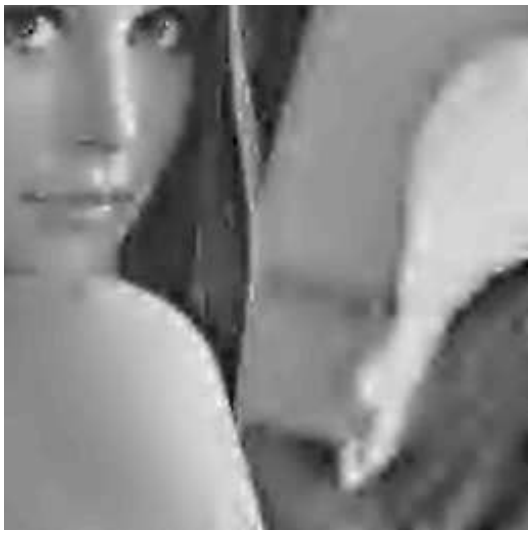
(AY), PSNR: 28,45, NNZ: 2802

(E1), PSNR: 28,23, NNZ: 2802

(E0), PSNR: 24,99, NNZ: 8883

CA_3, PSNR: 28,28, NNZ: 2719

Figura 2.22: *Imágenes resultantes después de aplicar el algoritmo de compresión con las distintas técnicas (AY), (E0) y (E1) para* lena.



(AY), PSNR: 28,45, NNZ: 2802

(E1), PSNR: 28,23, NNZ: 2802

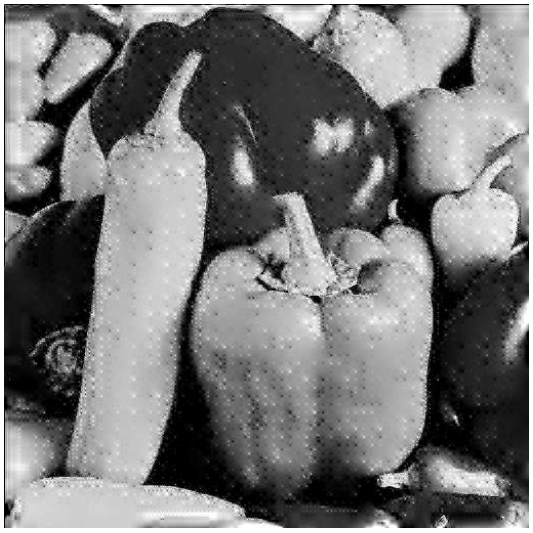
(E0), PSNR: 24,99, NNZ: 8883

CA_3, PSNR: 28,28, NNZ: 2719

Figura 2.23: *Ampliación de una zona de la imagen* lena *después de aplicar el algoritmo de compresión con las distintas técnicas (AY), (E0), (E1).*

(AY), PSNR: 28,01, NNZ: 4221

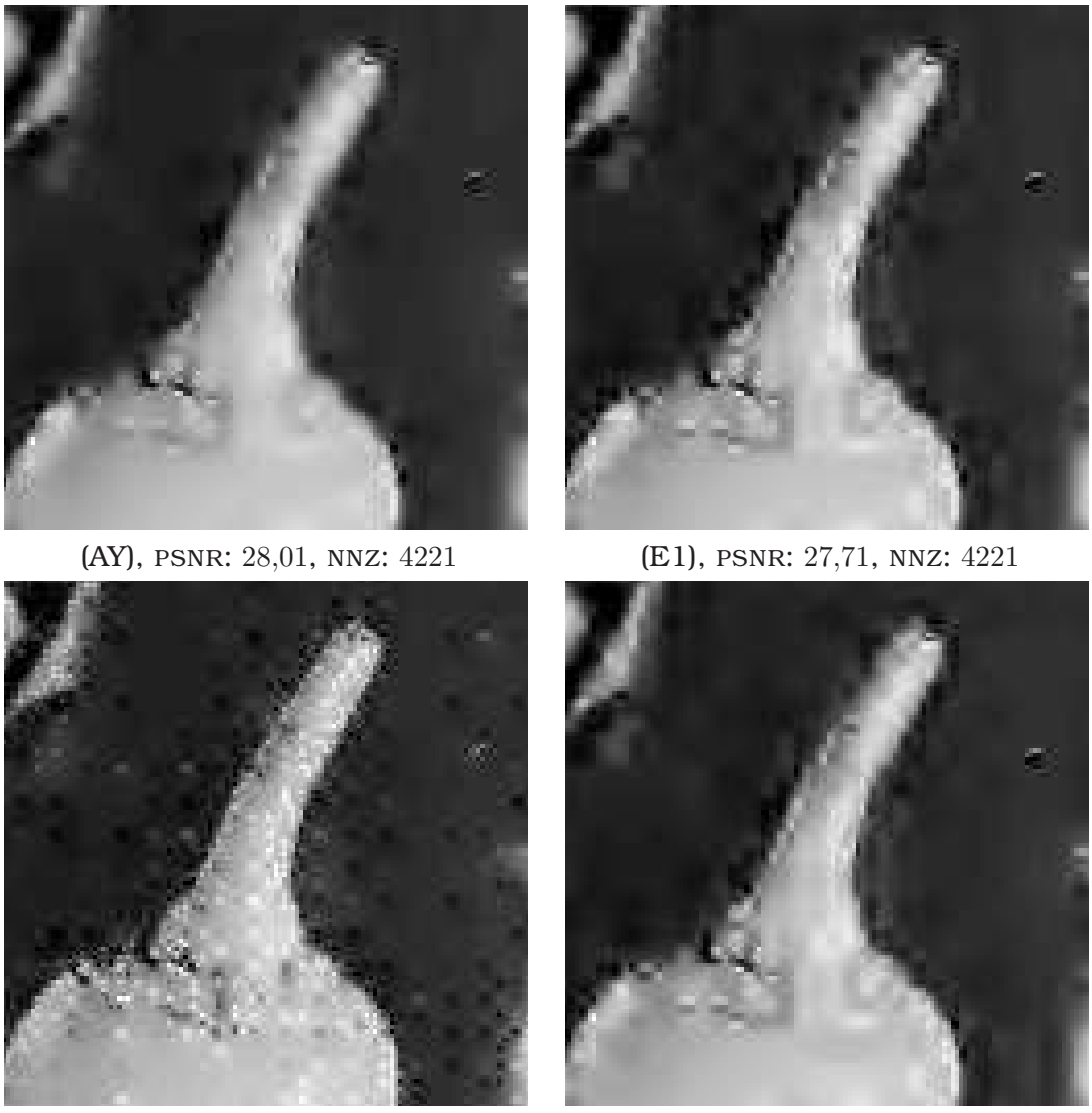
(E1), PSNR: 27,71, NNZ: 4221



(E0), PSNR: 24,89, NNZ: 12574

CA_3, PSNR: 27,77, NNZ: 4138

Figura 2.24: *Imágenes resultantes después de aplicar el algoritmo de compresión con las distintas técnicas (AY), (E0) y (E1) para* peppers.



(AY), PSNR: 28,01, NNZ: 4221 (E1), PSNR: 27,71, NNZ: 4221

(E0), PSNR: 24,89, NNZ: 12574 CA_3, PSNR: 27,77, NNZ: 4138

Figura 2.25: *Ampliación de una zona de la imagen* peppers *después de aplicar el algoritmo de compresión con las distintas técnicas (AY), (E0), (E1).*

2.5 Conclusiones y futuras líneas de investigación

En este capítulo hemos presentado la multiresolución *à la Harten*. Se basa en cuatro operadores fundamentales: discretización, $\mathcal{D}_k$ y reconstrucción $\mathcal{R}_k$, decimación $\mathcal{D}_k^{k-1}$ y predicción $\mathcal{P}_{k-1}^k$ cumpliendo una serie de características:

1. $\mathcal{D}_k^{k-1}$ tiene que ser lineal y sobreyectivo.
2. Relación de consistencia: $\mathcal{D}_k^{k-1}\mathcal{P}_{k-1}^k = I_{V^{k-1}}$.

En la literatura sobre multiresolución *à la Harten* el operador discretización proviene de la convolución de una función continua y una función B-*spline*. Sin embargo Getreuer y Meyer en [46] presentan otro tipo de discretizaciones (no *splines*) con operadores consistentes. Nosotros presentamos una estrategia que hemos denotado como (AY) que permite utilizar operadores predicción no consistentes y cuya principal característica es que conserva la naturaleza del operador hallado (los resultados por tanto serán mejores), el control del error y el orden. Sería interesante como posible línea de investigación considerar ambas teorías juntas: operador discretización no *splines* y operadores no consistentes.

En los próximos Capítulos 3 y 4 hallaremos operadores predicción que no son consistentes y veremos entonces la importancia de utilizar esta técnica.

Otra posible línea de investigación que se abre por medio de esta estrategia es la generalización de la utilización de medias PPH mencionadas en la §2.2.3 (ver para más detalle [5, 4]) para otros operadores decimación.

3

Operadores basados en técnicas estadísticas: Métodos núcleo

Los métodos núcleo se han desarrollado recientemente como parte de la teoría de métodos aditivos (ver p. ej. [55]). Desde finales del siglo XIX empezaron a trabajar en regresión local (caso específico de métodos núcleo en el que nos vamos a centrar) autores como Schiaparelli, De Forest o Gram, famoso por la técnica de ortogonalización de vectores de Gram-Schmidt ([79, 37, 48]). Es a comienzos del siglo XX cuando Spencer propone una fórmula para el resultado de la serie formada por los datos iniciales ya que la aproximación por regresión lineal no se ajusta correctamente a un problema sobre el índice de mortalidad ([82]). Sobre los años 30 y 40 se desarrollan métodos para resolver problemas de series temporales económicas. Por ejemplo el libro de Macaulay ([71]) que se basó en los trabajos realizados por Henderson ([60]) y otros muchos

([6, 67]). A comienzos de 1954 se desarrollan una serie de programas informáticos para el ajuste de series temporales bajo el nombre de método X-11 ([80]).

Hasta 1970 la regresión local no tiene mucha presencia dentro de la literatura estadística. Es en los años setenta cuando trabajos independientes de Stone [84, 85], Katkovnik [66] y el procedimiento LOWESS de Cleveland y Devlin [29] desarrollan de forma teórica estos métodos.

Una alternativa al tratamiento teórico sobre regresión fue ver el método como una extensión de métodos núcleo, por ejemplo el trabajo de Wand y Jones [92] y la extensión por Fan y Gijbels [43]. Hay muchos otros procedimientos para ajustar una curva desarrollados durante el siglo XX ([70, 56] y las referencias allí contenidas). Destacar también que las redes neuronales, tan de moda en los últimos años, se pueden ver como métodos núcleo ([27]).

3.1 Introducción

La regresión local (dentro de los métodos de núcleo) ha sido desarrollada en los últimos años dando solución a problemas de diferentes campos de la ciencia como economía, química, gráficos diseñados por ordenador, máquina de aprendizaje. Hay mucha literatura sobre estos temas (ver p. ej. [70, 56] y las referencias allí contenidas). Nosotros en este trabajo no vamos a entrar en la profundidad y en el desarrollo de la técnica, tan solo vamos a utilizar esta técnica para la construcción de un operador predicción dentro del esquema de multiresolución visto en el capítulo 2. La adaptación de este método a los datos iniciales hace que obtengamos buenos resultados. A su vez este método es lineal lo que nos permite probar sin dificultades la estabilidad. Una de las principales características que tendrá el método es que puede tomar una gran cantidad de puntos sin subir el grado del polinomio que aproxima. Este método generaliza los ya vistos utilizando interpolación ([15, 13, 73]) pues éstos son un caso particular de los métodos de núcleo.

En este capítulo comenzaremos repasando brevemente la regresión local sin entrar en mucho detalle en la teoría de la misma (ver [70] para más detalles). Después utilizando esta técnica construiremos un esquema de multiresolución, para ello fijaremos el operador discretización (valores puntuales, medias en celda y medias *hat*) y probaremos las propiedades que debe cumplir: orden y estabilidad. Hablaremos de la

posibilidad de hacerlo en varias dimensiones por medio de producto tensorial o directamente.

3.2 Regresión local polinómica

Vamos a describir regresión local en una dimensión. Más adelante también lo veremos en dos dimensiones. Supongamos que tenemos una función real $f(x)$, y de esta función un conjunto de n valores discretos (con nivel de resolución $k-1$, ver capítulo 2), $f_1^{k-1},\dots,f_n^{k-1}$, para cada valor $x_1^{k-1},\dots,x_n^{k-1}$ obtenidos por un proceso de discretización.

Nota 3.1. *En el caso de valores puntuales simplemente tenemos que* $f_j^{k-1} = (\mathcal{D}_{k-1}f)_j = f(x_j^{k-1})$.

Trataremos el caso de medias en celda de dos maneras: en primer lugar utilizaremos la Ecuación (2.29) que relaciona los valores de f_j^{k-1} *con los de su primitiva* F_j^{k-1} *y con éstos trabajaríamos la regresión local. Después generalizaremos la regresión local puntual a regresión local en medias en celda, esta generalización perderá la consistencia en el esquema de multiresolución, debido a esto utilizaremos la estrategia (AY) descrita en la §2.3.*

En el caso de medias hat *tan solo utilizamos la relación (2.36) que describe la biyección entre* f_j^{k-1} *con los de su segunda primitiva* H_j^{k-1} *y con éstos trabajaríamos la regresión local.*

Queremos ajustarla por un modelo simple en cada punto x_γ^k basándonos en aquellos puntos que estén cercanos a él y cuyo resultado global sea una función suave $\hat{z}(x)$. Esta localización se produce vía una función peso o núcleo $K_\lambda(x_\gamma^k, x_j^{k-1})$, que asigna un peso a cada x_j^{k-1} basado en la distancia de éste a x_γ^k.

Nota 3.2. *Como los valores donde se discretiza la función en los distintos niveles* k *están contenidos unos en otros, i. e.* $X^{k-1} \subset X^k$*, entonces no tenemos problemas para definir la distancia entre dos puntos de distintos niveles.*

Los núcleos K_λ son indexados por el parámetro λ que dictamina la anchura de la banda $(x_\gamma^k - \lambda, x_\gamma^k + \lambda)$; aquellos puntos que estén en esta banda serán los que utilicemos para la regresión local, así

$$K_\lambda(x_\gamma^k, x) = \omega\left(\frac{x_\gamma^k - x}{\lambda}\right) \tag{3.1}$$

donde $\omega(u) \geq 0$ es una función peso que asigna pesos a las observaciones cercanas a x_γ^k. Por ejemplo, podríamos utilizar la función tricubo que denotaremos como `tcub`:

$$\omega(x) = (1 - |x|^3)^3.$$

Supongamos que aproximamos $z(x)$ por un polinomio de grado r, entonces:

$$z(x) \approx \sum_{j=0}^{r} \beta_j x^j. \tag{3.2}$$

Sea $L(x, y)$ una función de pérdida, (p. ej. $L(x, y) = (x - y)^2$) entonces nuestro problema es el siguiente:

$$\begin{aligned} \hat{z}(x) &= \operatorname*{arg\,mín}_{z(x)\in\Pi_1^r(\mathbb{R})} \sum_{j=1}^{n} K_\lambda(x_\gamma^k, x_j^{k-1}) L(f_j^{k-1}, z(x_j^{k-1})) \\ (\hat{\beta}_0, \ldots, \hat{\beta}_r) &= \operatorname*{arg\,mín}_{\beta_i\in\mathbb{R}, i=0,\ldots,r} \sum_{j=1}^{n} K_\lambda(x_\gamma^k, x_j^{k-1}) L(f_j^{k-1}, \sum_{i=0}^{r} \beta_i (x_j^{k-1})^i) \end{aligned} \tag{3.3}$$

cuya solución es $\hat{z}(x_\gamma^k) = \sum_{i=0}^{r} \hat{\beta}_i (x_\gamma^k)^i$.

Veamos un primer ejemplo sencillo.

Ejemplo 3.1. *Supongamos que tomamos un polinomio de grado* 0*, es decir, una constante* β*, y nuestra función pérdida* $L(x, y) = (x - y)^2$*; así el problema es:*

$$\operatorname*{arg\,mín}_{\beta\in\mathbb{R}} \sum_{j=1}^{n} K_\lambda(x_\gamma^k, x_j^{k-1}) \big(f_j^{k-1} - \beta\big)^2.$$

El mínimo en este caso es

$$\hat{\beta} = \frac{\sum_{j=1}^{n} K_\lambda(x_\gamma^k, x_j^{k-1}) f_j^{k-1}}{\sum_{j=1}^{n} K_\lambda(x_\gamma^k, x_j^{k-1})},$$

si además, suponemos que el núcleo es la función

$$K_\lambda(x_\gamma^k, x) = \chi_{[-1,1]}\left(\frac{x_\gamma^k - x}{\lambda}\right)$$

entonces tenemos que

$$\hat{\beta} = \frac{1}{|\mathcal{S}|} \sum_{j\in\mathcal{S}} f_j^{k-1},$$

siendo $\mathcal{S} = \{i \,|\, 1 \leq i \leq n, |x_i^{k-1} - x_\gamma^k| \leq \lambda\}$.

Obtenemos el método que se denomina el vecino más cercano ([56]).

Ejemplo 3.2. *Supongamos un mallado como el descrito para la multiresolución utilizando valores puntuales $X^{k-1} \subset X^k$. Y supongamos que tenemos los $n = 2^{k-1} + 1$ pares $(x_1^{k-1}, f_1^{k-1}), \ldots, (x_n^{k-1}, f_n^{k-1})$, con $x_j^{k-1} - x_{j-1}^{k-1} = h_{k-1}$, $\forall\, j = 2, \ldots, n$. Sea $x_\gamma^k \in X^k \setminus X^{k-1}$ un punto intermedio. Tomamos un valor λ tal que existen cuatro puntos, $x_{\gamma-2}^{k-1}, x_{\gamma-1}^{k-1}, x_{\gamma 0}^{k-1}, x_{\gamma 1}^{k-1} \in X^{k-1}$ que están en la banda $(x_\gamma^k - \lambda, x_\gamma^k + \lambda)$. Supongamos que tenemos el mismo núcleo que en el ejemplo anterior (Ejemplo 3.1) y la misma función de pérdida, es decir, $K_\lambda(x_\gamma^k, x) = \chi_{[-1,1]}\big(\frac{x_\gamma^k - x}{\lambda}\big)$ y $L(x, y) = (x - y)^2$.*

Tomamos una aproximación de grado $r = 3$, así tenemos el problema como:

$$(\hat\beta_0, \ldots, \hat\beta_3) = \underset{\beta_i \in \mathbb{R}, i=0,\ldots,3}{\arg\min} \sum_{j=1}^{n} K_\lambda(x_\gamma^k, x_j^{k-1}) \big(f_j^{k-1} - \sum_{i=0}^{3} \beta_i (x_j^{k-1})^i\big)^2$$

$$(\hat\beta_0, \ldots, \hat\beta_3) = \underset{\beta_i, i=0,\ldots,3}{\arg\min} \sum_{j=-2}^{1} \big(f_{\gamma_j}^{k-1} - [\beta_0 + \beta_1 (x_{\gamma_j}^{k-1}) + \beta_2 (x_{\gamma_j}^{k-1})^2 + \beta_3 (x_{\gamma_j}^{k-1})^3]\big)^2, \tag{3.4}$$

cuya solución es

$$\hat z(x_\gamma^k) = -\frac{1}{16} f_{\gamma_{-2}}^{k-1} + \frac{9}{16} f_{\gamma_{-1}}^{k-1} + \frac{9}{16} f_{\gamma_0}^{k-1} - \frac{1}{16} f_{\gamma_1}^{k-1},$$

que es el operador predicción hallado para valores puntuales utilizando interpolación polinómica de grado 3 (§2.2.3, Ecuación (2.24)).

Por tanto, hemos visto por los dos ejemplos anteriores que este método generaliza tanto la técnica del vecino más próximo como la interpolación polinómica a trozos, difiriendo de ésta última en que al aproximar asigna un peso determinado a los valores dependiendo de la lejanía con el punto a interpolar (en el caso de interpolación polinómica todos los valores tienen el mismo peso).

Veamos cuáles son las componentes de la regresión local. Las analizaremos bajo el prisma de las propiedades que podemos utilizar en multiresolución (para más detalles ver [70]).

3.3 Componentes de la regresión local

En el problema de regresión local tenemos que especificar un conjunto de variables: la anchura de la banda, el grado del polinomio local,

la función peso y la función de pérdida. Vamos a analizar estas componentes sin entrar en un excesivo detalle de las propiedades estadísticas (para más detalles consultar [70] y el capítulo 6 de [56]).

3.3.1 Anchura de la banda

La banda λ tiene un efecto de gran importancia en la regresión local. Si λ es demasiado pequeña, tenemos insuficientes datos en la ventana de interpolación (lo que equivaldría en interpolación segmentaria a tomar una *stencil* con pocos puntos). Si, por el contrario, la banda es demasiado amplia entonces podemos encontrarnos con alguna discontinuidad que reduzca el orden de aproximación.

Muchos autores (ver [70]) utilizan una banda dependiente del punto donde interpolan, i. e. $\lambda(x_\gamma^k)$, sin embargo, en nuestro caso, como vamos a utilizar mallas de distancia fija entre los puntos, es decir, mallas equiespaciadas, entonces será suficiente con tomar una anchura de banda constante, λ.

Nota 3.3. *En multiresolución tomaremos la anchura de la banda dependiendo del nivel de multiresolución, lo denotaremos como λ_k. Por comodidad, siempre que en un ejemplo hagamos un solo paso de la multiresolución omitiremos la referencia al nivel.*

Si la malla no fuese uniforme podríamos incurrir en error al no tomar ningún punto para la regresión. Para más detalle en la elección de λ en mallas no uniformes ver p. ej. [70] o la §6 de [56].

En el siguiente ejemplo sencillo vemos los distintos resultados que obtenemos con diferentes valores λ.

Ejemplo 3.3. *Tomamos un mallado equiespaciado de 128 puntos contenido en el intervalo cerrado $[-1,1]$ con $h = 1/64$, $x_j = -1 + jh$ con $j = 1,\ldots,128$, y sea $f(x) = e^{x^2} + x^3 + \epsilon$, donde ϵ es un ruido blanco gaussiano. Así $f_j = f(x_j)$. Hallamos el conjunto de 256 puntos $\{f(y_j)\}_{j=1}^{256}$ con $y_j = -1 + j\frac{h}{2}$ con $j = 1,\ldots,256$ utilizando como función núcleo $\omega(x) = 1 - x^2$, $|x| < 1$ que llamaremos núcleo* `epan` *como veremos en la §3.3.3, como grado del polinomio que aproxima $r = 2$ y $L(x,y) = (x-y)^2$. Tomamos $\lambda = 0{,}046$ y $\lambda = 0{,}406$ que dada la distancia entre los puntos del mallado involucra a 6 y 52 puntos respectivamente. Como podemos ver en la Figura 3.1 tomando un λ pequeño encontramos grandes problemas debido al ruido introducido. Si tomamos un λ mayor vemos que se diluye este ruido obteniendo (en este caso) mejores resultados.*

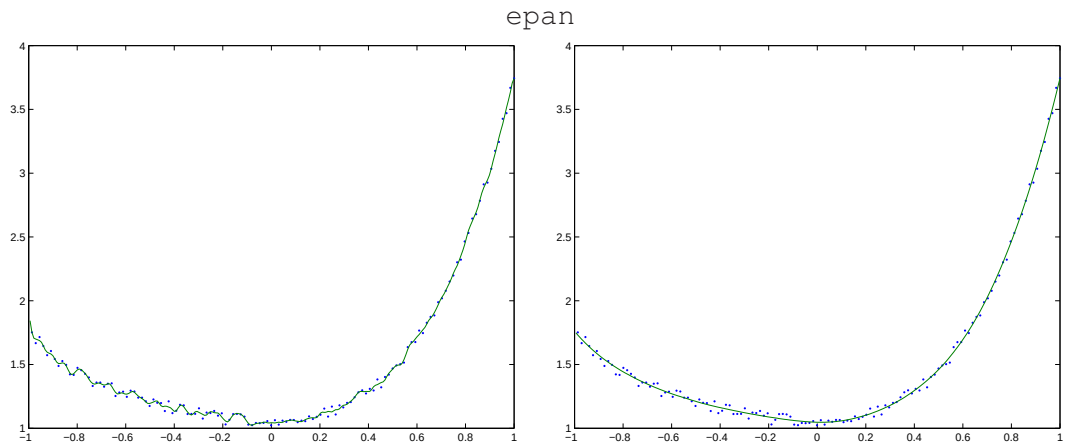


Figura 3.1: *Ejemplo tomando diferentes valores λ con función peso* `epan` *y con grado del polinomio $r = 2$.*

3.3.2 El grado del polinomio

El polinomio que tomamos es

$$z(x) = \sum_{j=0}^{r} \beta_j x^j = \beta^T A(x) \tag{3.5}$$

donde $\beta = (\beta_0, \ldots, \beta_r)^T$ y $A(x) = (1, x, x^2, \ldots, x^r)^T$, siempre que $|x - x_\gamma^k| < \lambda$. Como en la elección de la anchura de banda, el grado del polinomio es de gran importancia para el desarrollo de la aproximación. Un alto grado del polinomio siempre da una mejor aproximación que un polinomio de grado menor. La diferencia fundamental con la interpolación polinómica a trozos es que el grado del polinomio r es independiente del número de puntos que se tome. En interpolación polinómica si tomas t puntos el grado del polinomio es $t - 1$, esto es evidente para poder resolver el sistema que aparece (la solución existe y es única, ver p. ej. [21, 22]). En el caso de regresión local podemos fijar el grado del polinomio r y tomar un número mayor de puntos $s > r$.

Ejemplo 3.4. *Tomamos el mismo ejemplo que en la sección anterior (Ej. 3.3) pero en este caso fijamos $\lambda = 0{,}4$ y tomamos como parámetro el grado*

del polinomio r. Así obtenemos, como vemos en la Figura 3.2, dos curvas diferentes tomando $r = 1$ y $r = 5$, siendo la segunda la que mejor aproxima a la función real $f(x) = e^{x^2} + x^3$.

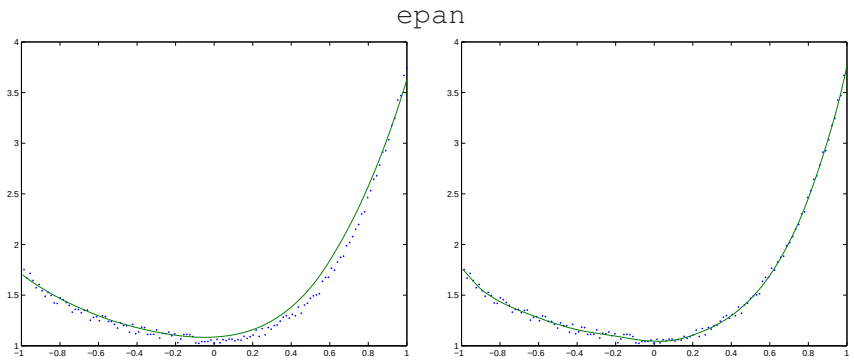


Figura 3.2: *Ejemplo tomando diferentes grados del polinomio que aproxima r con función peso* `epan` *y con grado del polinomio $\lambda = 0{,}4$.*

3.3.3 La función peso

La función peso influye de forma directa en la calidad visual y numérica de la curva de regresión. La forma de la función de peso es:

$$K_\lambda(x, x_\gamma^k) = \omega\left(\frac{x - x_\gamma^k}{\lambda}\right)$$

donde $\omega(u) \geq 0$ es una función que suele ser simétrica y continua (como veremos a continuación), exceptuando el caso $\omega(x) = \chi_{[-1,1]}(x)$. Hay distintas funciones peso que podemos utilizar, la más sencilla es $K_\lambda(x, x_\gamma^k) = \chi_{[-1,1]}\left(\frac{x - x_\gamma^k}{\lambda}\right)$. Se pueden tomar otras muchas funciones como la función tricubo (comúnmente seleccionada). En la literatura de regresión local existen otros tipos de funciones peso, p. ej. funciones peso hacia un lado, (ver §3.5 o §6.3 de [70]). En este trabajo no vamos a entrar a estudiar otras posibles funciones que las ya utilizadas comúnmente en la literatura. Veamos un cuadro de los pesos que vamos a utilizar con la notación propuesta en [70]:

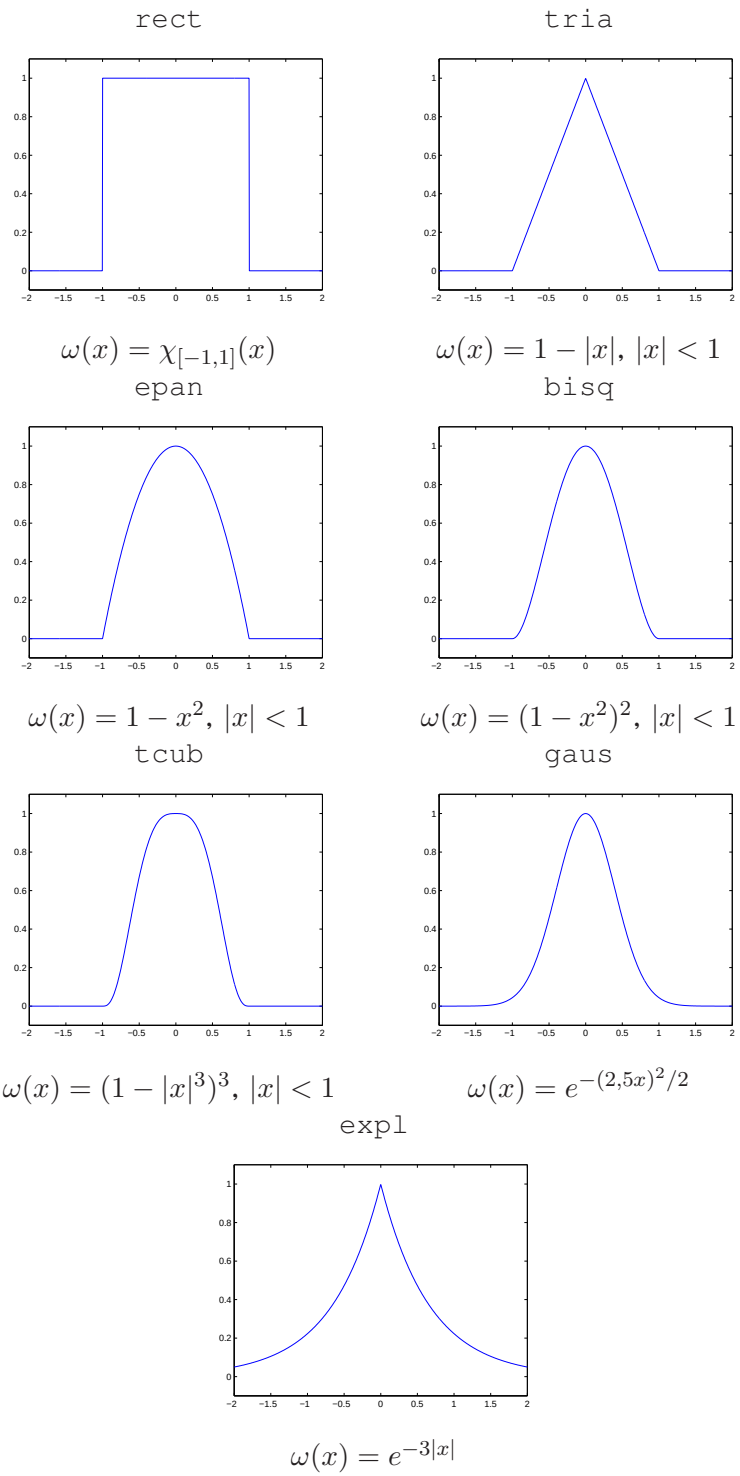


Figura 3.3: *Funciones peso. La función* `epan` *corresponde a la estudiada por Epanechnikov en [42].*

3.3.4
Función de pérdida y linealidad del operador

La función de pérdida marca la distancia entre los elementos de aprendizaje que ya conocemos, en nuestro caso la distancia entre los valores en el nivel $k-1$: f^{k-1} y la aproximación polinómica en cada uno de los valores del mallado en el nivel $k-1$, $x^{k-1} \in X^{k-1}$. Utilizaremos la norma ℓ^2 (en la §3.6 haremos una breve introducción al uso de otras normas), es decir,

$$L(x,y) = |x-y|^2. \tag{3.6}$$

Tenemos nuestro problema visto en la Ecuación (3.3) como:

$$\hat{\beta} = \underset{\beta_i \in \mathbb{R}, i=0,\dots,r}{\arg\min} \sum_{j=1}^{n} K_\lambda(x_\gamma^k, x_j^{k-1}) \big(f_j^{k-1} - \sum_{i=0}^{r} \beta_i (x_j^{k-1})^i\big)^2 \tag{3.7}$$

Nota 3.4. *Suponemos que todos los elementos del muestreo, x_j^{k-1}, $j = 1,\dots,n$ tienen un peso estrictamente positivo respecto del valor a x_γ^k, es decir:*

$$K_\lambda(x_\gamma^k, x_j^{k-1}) > 0, \quad \forall j = 1,\dots,n.$$

Si tuviésemos valores con peso cero entonces reduciríamos el problema (3.3) a los valores estrictamente positivos únicamente. Así supongamos $1 \le s < t \le n$ con $s+t \ge r$ tal que

$$K_\lambda(x_\gamma^k, x_j^{k-1}) > 0, \quad \forall j = s,\dots,t,$$

entonces el problema quedaría como

$$\hat{\beta} = \underset{\beta_i \in \mathbb{R}, i=0,\dots,r}{\arg\min} \sum_{j=s}^{t} K_\lambda(x_\gamma^k, x_j^{k-1}) \big(f_j^{k-1} - \sum_{i=0}^{r} \beta_i (x_j^{k-1})^i\big)^2,$$

y se resolvería de la misma manera que vamos a ver a continuación.

Sea $\beta = (\beta_0,\dots,\beta_r)^T$ y $f^{k-1} = (f_1^{k-1},\dots,f_n^{k-1})^T$, sea la matriz $n \times (r+1)$

$$\mathbf{X} = \begin{pmatrix} 1 & x_1^{k-1} & \dots & (x_1^{k-1})^r \\ 1 & x_2^{k-1} & \dots & (x_2^{k-1})^r \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_n^{k-1} & \dots & (x_n^{k-1})^r \end{pmatrix}; \tag{3.8}$$

y la matriz $n \times n$

$$\mathbf{W}_\lambda(x_\gamma^k) = \begin{pmatrix} \omega\left(\frac{x_1^{k-1}-x_\gamma^k}{\lambda}\right) & 0 & \dots & 0 \\ 0 & \omega\left(\frac{x_2^{k-1}-x_\gamma^k}{\lambda}\right) & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \omega\left(\frac{x_n^{k-1}-x_\gamma^k}{\lambda}\right) \end{pmatrix} \tag{3.9}$$

entonces podemos ver el problema para $p = 2$ como la búsqueda de la solución del siguiente sistema

$$\mathbf{X}^T\mathbf{W}_\lambda(x_\gamma^k)\mathbf{X}\beta = \mathbf{X}^T\mathbf{W}_\lambda(x_\gamma^k)f^{k-1}, \tag{3.10}$$

que, si $|\mathbf{X}^T\mathbf{W}_\lambda(x_\gamma^k)\mathbf{X}| \neq 0$, sería

$$\hat{\beta} = (\mathbf{X}^T\mathbf{W}_\lambda(x_\gamma^k)\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}_\lambda(x_\gamma^k)f^{k-1}.$$

Así pues nuestro operador es $\hat{z}(x) = A(x)^T\hat{\beta}$ y nuestra aproximación en el punto x_γ^k es

$$\begin{aligned} \hat{z}(x_\gamma^k) &= A(x_\gamma^k)^T(\mathbf{X}^T\mathbf{W}_\lambda(x_\gamma^k)\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}_\lambda(x_\gamma^k)f^{k-1} \\ &= \sum_{j=1}^{n} l_j(x_\gamma^k)f_j^{k-1}, \end{aligned} \tag{3.11}$$

donde $l_j(x_\gamma^k) = A(x_\gamma^k)^T(\mathbf{X}^T\mathbf{W}_\lambda(x_\gamma^k)\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}_\lambda(x_\gamma^k)e_j$, con $e_j = (\delta_{i,j})_{i=1}^n$ que no depende de f^{k-1} y por tanto nuestro operador es lineal.

Si tomamos el Ejemplo 3.1 donde se aproximaba por una constante teníamos que

$$\hat{\beta} = \frac{\sum_{j=1}^n K_\lambda(x_\gamma^k, x_j^{k-1})f_j^{k-1}}{\sum_{j=1}^n K_\lambda(x_\gamma^k, x_j^{k-1})},$$

por tanto,

$$l_j(x_\gamma^k) = \frac{K_\lambda(x_\gamma^k, x_j^{k-1})}{\sum_{i=1}^n K_\lambda(x_\gamma^k, x_i^{k-1})}.$$

Como podemos ver el **operador es lineal** independientemente de la función peso que tomemos, esto es de gran importancia para probar la estabilidad del esquema de multiresolución determinada en la Definición 2.3 de la §2.2.1.

3.4 Esquema de multiresolución basado en regresión local

Vamos a diseñar un esquema de multiresolución. Para ello tendremos que definir dos operadores como hemos visto en el capítulo 2: el operador decimación que será uno de los operadores ya vistos basado en la función *spline*, es decir, decimación en valores puntuales, medias en celda o medias *hat*; y el operador predicción, $\mathcal{P}_{k-1}^{k}$, que lo definimos utilizando regresión local. Calcularemos el orden de aproximación con independencia de la función peso elegida, daremos algunos ejemplos y veremos la estabilidad del esquema de multiresolución.

3.4.1 Multiresolución para valores puntuales

Como veíamos en la §2.2.2 uno de los operadores que podíamos utilizar era aquel cuyas entradas son los valores de una cierta función en una malla. Así definíamos, $f_j^k = (\mathcal{D}_k f)_j = f(x_j^k)$. Y nuestro operador decimación era

$$f_j^{k-1} = (\mathcal{D}_k^{k-1} f^k)_j = f_{2j}^k, \quad j = 0, \dots, J_{k-1}. \tag{3.12}$$

Por tanto, debemos definir un operador predicción para los puntos x_{2j-1}^k con $j = 1, \dots, J_{k-1}$ utilizando las parejas de valores $(x_j^{k-1}, f_j^{k-1})_{j=0}^{J_{k-1}}$. Aplicamos regresión local con este conjunto de valores. Con la notación vista en la sección anterior definimos el operador predicción como:

$$\begin{cases} (\mathcal{P}_{k-1}^k f^{k-1})_{2j} = f_j^{k-1}, & j = 0, \dots, J_{k-1}; \\ (\mathcal{P}_{k-1}^k f^{k-1})_{2j-1} = l(x_{2j-1}^k) f^{k-1}, & j = 1, \dots, J_{k-1}, \end{cases} \tag{3.13}$$

donde

$$l(x_{2j-1}^k) = A(x_{2j-1}^k)^T (\mathbf{X}^T \mathbf{W}_\lambda(x_{2j-1}^k)\mathbf{X})^{-1} \mathbf{X}^T \mathbf{W}_\lambda(x_{2j-1}^k). \tag{3.14}$$

En este caso tenemos un operador lineal, además como los puntos que conforman nuestro mallado están a la misma distancia no tendremos que resolver la ecuación en cada punto sino que obtendremos unos filtros para todos los puntos en cualquier nivel de resolución k dependiendo de la función peso que utilicemos K_λ y del valor del ancho

de banda λ. En la §3.4.2 podemos encontrar algunos ejemplos de estos filtros obtenidos teniendo en cuenta la función peso, el ancho de banda y el grado del polinomio que elegimos.

3.4.2

Filtros uno dimensionales obtenidos en multiresolución utilizando regresión local lineal

En esta sección mostramos las tablas con los filtros obtenidos utilizando la multiresolución para los métodos `rect` y `gaus`. También mostramos las funciones límite de $f = (\delta_{l,0})_{l\in\mathbb{Z}}$, Figuras 3.4 y 3.5.

rect	$r = 0, 1$				
λ	0,002	0,004	0,006	0,008	0,01
$j-5$					$\frac{1}{10}$
$j-4$				$\frac{1}{8}$	$\frac{1}{10}$
$j-3$			$\frac{1}{6}$	$\frac{1}{8}$	$\frac{1}{10}$
$j-2$		$\frac{1}{4}$	$\frac{1}{6}$	$\frac{1}{8}$	$\frac{1}{10}$
$j-1$	$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{6}$	$\frac{1}{8}$	$\frac{1}{10}$
j	$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{6}$	$\frac{1}{8}$	$\frac{1}{10}$
$j+1$		$\frac{1}{4}$	$\frac{1}{6}$	$\frac{1}{8}$	$\frac{1}{10}$
$j+2$			$\frac{1}{6}$	$\frac{1}{8}$	$\frac{1}{10}$
$j+3$				$\frac{1}{8}$	$\frac{1}{10}$
$j+4$					$\frac{1}{10}$
	$r = 2, 3$				
λ		0,004	0,006	0,008	0,01
$j-5$					$-\frac{14}{160}$
$j-4$				$-\frac{3}{32}$	$\frac{6}{160}$
$j-3$			$-\frac{3}{32}$	$\frac{3}{32}$	$\frac{21}{160}$
$j-2$		$-\frac{1}{16}$	$\frac{7}{32}$	$\frac{7}{32}$	$\frac{31}{160}$
$j-1$		$\frac{9}{16}$	$\frac{3}{8}$	$\frac{9}{32}$	$\frac{36}{160}$
j		$\frac{9}{16}$	$\frac{3}{8}$	$\frac{9}{32}$	$\frac{36}{160}$
$j+1$		$-\frac{1}{16}$	$\frac{7}{32}$	$\frac{7}{32}$	$\frac{31}{160}$
$j+2$			$-\frac{3}{32}$	$\frac{3}{32}$	$\frac{21}{160}$
$j+3$				$-\frac{3}{32}$	$\frac{6}{160}$
$j+4$					$-\frac{14}{160}$
	$r = 4, 5$				
λ			0,006	0,008	0,01
$j-5$					0,0390625
$j-4$				0,029296875	−0,1171875
$j-3$			$\frac{5}{256}$	−0,134765625	0,01171875
$j-2$			$-\frac{25}{256}$	0,166015625	0,21484375
$j-1$			$\frac{150}{256}$	0,439453125	0,3515625
j			$\frac{150}{256}$	0,439453125	0,3515625
$j+1$			$-\frac{25}{256}$	0,166015625	0,21484375
$j+2$			$\frac{5}{256}$	−0,134765625	0,01171875
$j+3$				0,029296875	−0,1171875
$j+4$					0,0390625

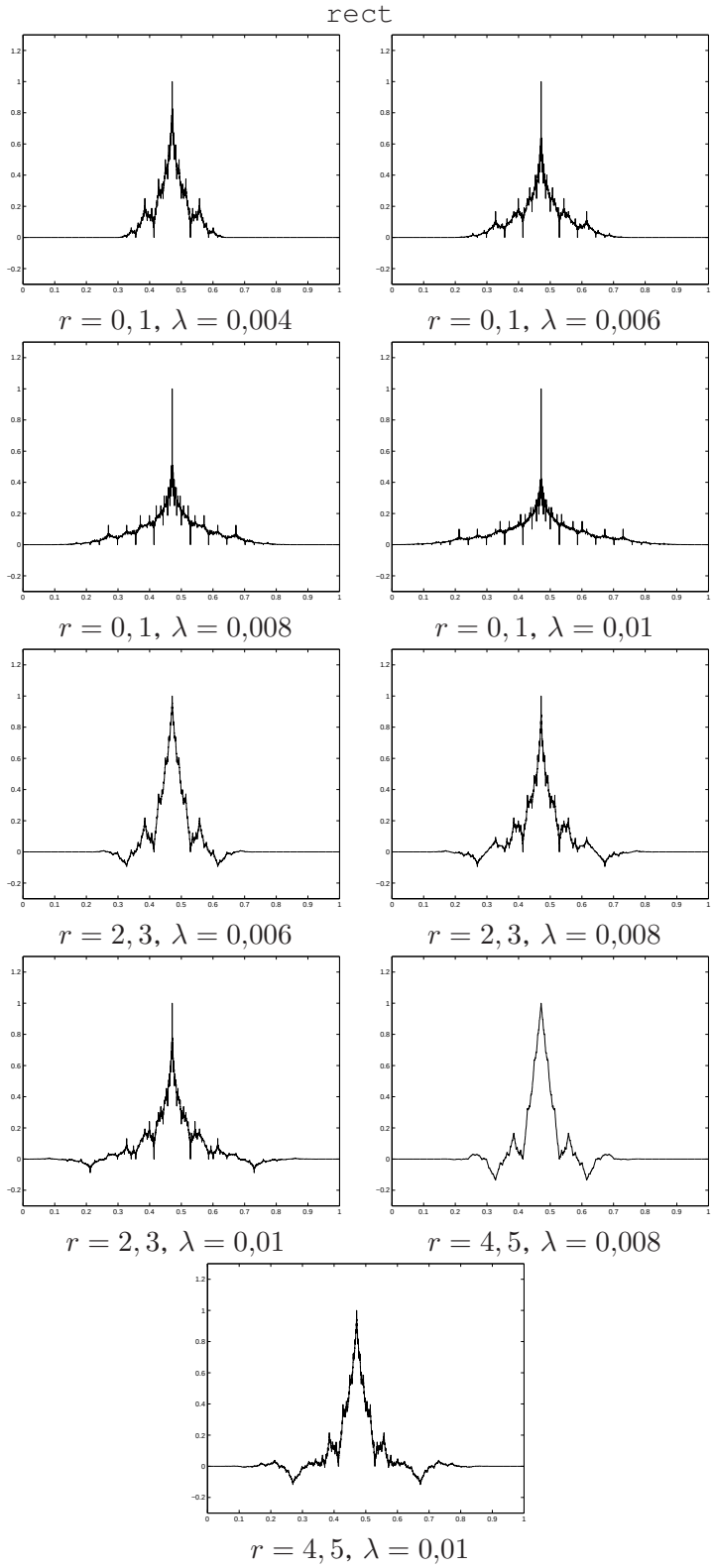


Figura 3.4: *Funciones límite para la función* $f = (\delta_{l,0})_{l \in \mathbb{Z}}$.

gaus			$r = 0, 1$		
λ	0,001	0,002	0,003	0,0035	0,004
$j-5$					0,000000136501029
$j-4$				0,000003701285787	0,000052933761585
$j-3$			0,000165078381162	0,001270927991390	0,004625731079355
$j-2$		0,001286040638636	0,033011492755959	0,062320820397541	0,091091697905835
$j-1$	$\frac{1}{2}$	0,498713959361364	0,466823428862879	0,436404550325282	0,404229500752196
j	$\frac{1}{2}$	0,498713959361364	0,466823428862879	0,436404550325282	0,404229500752196
$j+1$		0,001286040638636	0,033011492755959	0,062320820397541	0,091091697905835
$j+2$			0,000165078381162	0,097355349861664	0,004625731079355
$j+3$				0,001270927991390	0,000052933761585
$j+4$					0,000000136501029
			$r = 2, 3$		
λ		0,002	0,003	0,0035	0,004
$j-5$					−0,000001856975886
$j-4$				−0,000039193688584	−0,000404245234416
$j-3$			−0,001276200518393	−0,005927532956843	−0,014623486990411
$j-2$		$-\frac{1}{16}$	−0,058671398444821	−0,044482238997964	−0,016185497863416
$j-1$		$\frac{9}{16}$	0,559947598963214	0,550448965643392	0,531215087064129
j		$\frac{9}{16}$	0,559947598963214	0,550448965643392	0,531215087064129
$j+1$		$-\frac{1}{16}$	−0,058671398444821	−0,044482238997964	−0,016185497863416
$j+2$			−0,001276200518393	−0,005927532956843	−0,014623486990411
$j+3$				−0,000039193688584	−0,000404245234416
$j+4$					−0,014623486990411
			$r = 4, 5$		
λ			0,003	0,0035	0,004
$j-5$					0,000011205715821
$j-4$				0,000229802058785	0,001162101371773
$j-3$			$\frac{3}{256}$	0,010569739706076	0,005740157403820
$j-2$			$-\frac{25}{256}$	−0,095588031470938	−0,086805137600309
$j-1$			$\frac{150}{256}$	0,584788489706076	0,579891673108894
j			$\frac{150}{256}$	0,584788489706076	0,579891673108894
$j+1$			$-\frac{25}{256}$	−0,095588031470938	−0,086805137600309
$j+2$			$\frac{3}{256}$	0,010569739706076	0,005740157403820
$j+3$				0,000229802058785	0,001162101371773
$j+4$					0,000011205715821

3.4.3

Multiresolución para medias en celda

Para construir el método de multiresolución en el contexto de medias en celda planteamos dos alternativas. Por un lado si tenemos calculado el valor del operador predicción en el contexto de valores puntuales, utilizando la reconstrucción vía función primitiva es relativamente sencillo obtener la predicción en medias en celda en una dimensión como veremos a continuación. Sin embargo, utilizar esta misma estrategia en dos dimensiones es complicado. Así pues planteamos la generalización natural del método de multiresolución de núcleo definiendo una distancia entre celdas y diseñando un problema acorde con el operador decimación para medias en celdas.

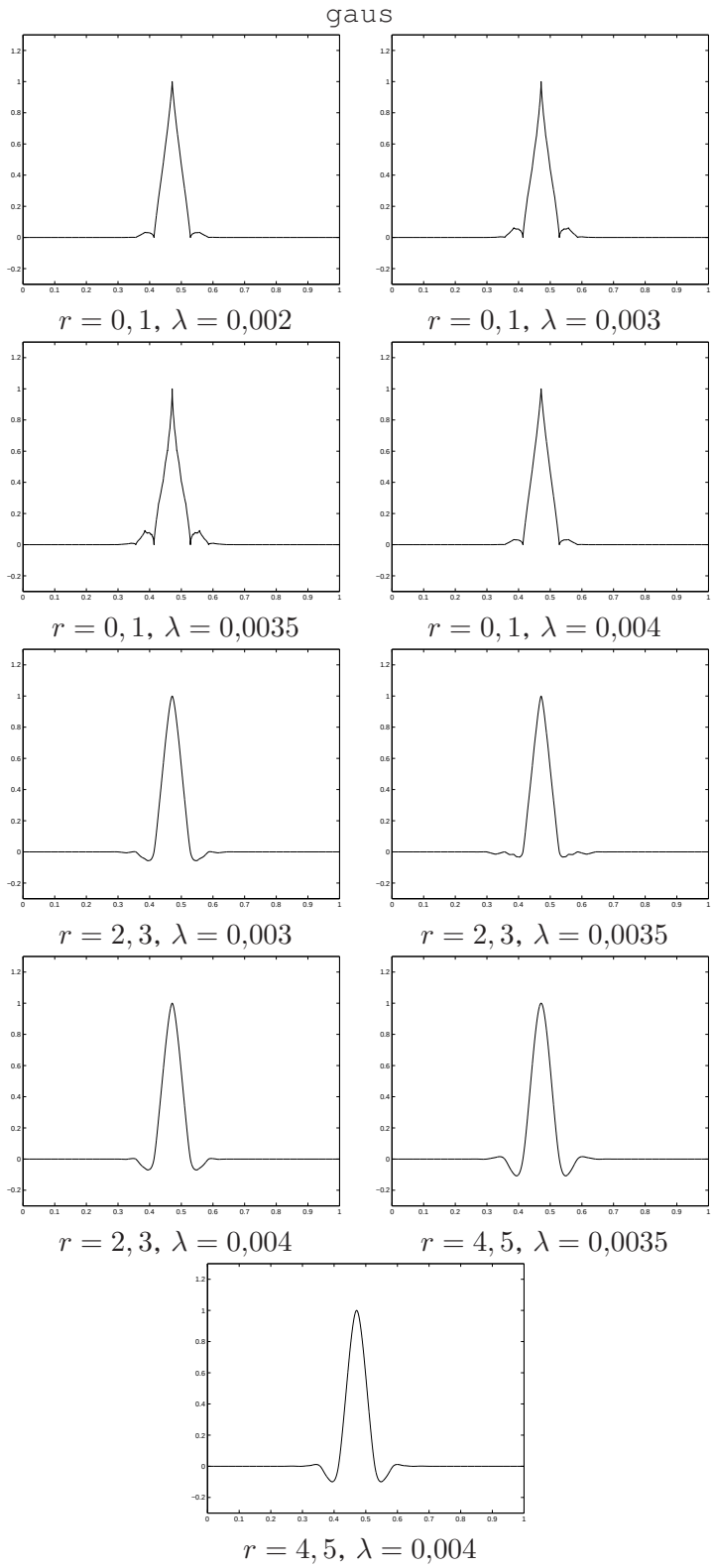


Figura 3.5: *Funciones límite para la función* $f = (\delta_{l,0})_{l \in \mathbb{Z}}$.

Primera alternativa: Cálculo del operador predicción en medias en celda basándonos en la predicción en valores puntuales

Al igual que veíamos en la §2.2.2 los valores originales son la media de una determinada función en una celda, así:

$$f_j^k = 2^k \int_{c_j^k} f(x)dx,$$

con $c_j^k = [(j-1)2^{-k}, j2^{-k}]$ y $j = 1, \ldots, 2^k$.

Por tanto se formulará utilizando la reconstrucción vía función primitiva que vimos en la §2.2.3 (ver también [52, 53, 13, 46]), es decir, llamaremos:

$$F(x) = \int_0^x f(y)dy, \quad f(x) = \frac{d}{dx}F(x).$$

Entonces teníamos la relación entre $\{f_j^k\}$ y $\{F_j^k\}$ siguiente

$$F_j^k = 2^{-k}\sum_{n=0}^{j} f_n^k, \quad f_j^k = 2^k(F_j^k - F_{j-1}^k), \quad j = 1, \ldots, J_k, \tag{3.15}$$

lo que nos permite definir el operador predicción en medias en celda en base al operador definido para valores puntuales de la función $F(x)$ como

$$(\mathcal{P}_{k-1}^k f^{k-1})_j = 2^k\big(\mathcal{I}(x_j^k; F^{k-1}) - \mathcal{I}(x_{j-1}^k; F^{k-1})\big), \tag{3.16}$$

donde $\mathcal{I}(x_\gamma^k, F^{k-1})$ representa la aproximación en el punto x_γ^k utilizando los valores $\{F^k\}$ como vimos en el capítulo anterior. Veamos un ejemplo donde queda reflejado cómo hallar los filtros en el caso de medias en celda basándonos en los filtros hallados para valores puntuales.

Ejemplo 3.5. *Supongamos que deseamos calcular el operador centrado utilizando tres celdas (es decir cuatro puntos). Para ello tenemos que:*

$$\begin{cases} \mathcal{I}(x_{2j-1}^k; F^{k-1}) = a_{-2}F_{j-2}^{k-1} + a_{-1}F_{j-1}^{k-1} + a_0F_j^{k-1} + a_1F_{j+1}^{k-1}, \\ \mathcal{I}(x_{2j-2}^k; F^{k-1}) = F_{j-1}^{k-1}. \end{cases}$$

con $a_i \in \mathbb{R}$, $i = -2, -1, 0, 1$ *y* $a_{-2} + a_{-1} + a_0 + a_1 = 1$. *Nuestra intención es hallar un operador predicción de la forma:*

$$\begin{cases} (\mathcal{P}_{k-1}^k f^{k-1})_{2j-1} = b_{-1}f_{j-1}^{k-1} + b_0f_j^{k-1} + b_1f_{j+1}^{k-1}, \\ (\mathcal{P}_{k-1}^k f^{k-1})_{2j} = \hat{b}_{-1}f_{j-1}^{k-1} + \hat{b}_0f_j^{k-1} + \hat{b}_1f_{j+1}^{k-1}. \end{cases}$$

cumpliendo que $(\mathcal{P}_{k-1}^k f^{k-1})_{2j-1} + (\mathcal{P}_{k-1}^k f^{k-1})_{2j} = 2f_j^{k-1}$. *Así tenemos que:*

$$\begin{aligned} b_{-1} &= -\hat{b}_{-1}, \\ b_0 &= 2 - \hat{b}_0, \\ b_1 &= -\hat{b}_1. \end{aligned}$$

Así, utilizando las Ecuaciones (3.15) y (3.16) podemos definir nuestro operador como:

$$\begin{aligned} (\mathcal{P}_{k-1}^k f^{k-1})_{2j-1} &= 2^k\big(\mathcal{I}(x_{2j-1}^k; F^{k-1}) - \mathcal{I}(x_{2j-2}^k; F^{k-1})\big) \\ &= 2^k\big(\mathcal{I}(x_{2j-1}^k; F^{k-1}) - \mathcal{I}(x_{2j-2}^k; F^{k-1})\big) \\ &= 2^k(a_{-2}F_{j-2}^{k-1} + a_{-1}F_{j-1}^{k-1} + a_0F_j^{k-1} + a_1F_{j+1}^{k-1} - F_{j-1}^{k-1}) \\ &= 2(a_{-2} + a_{-1} + a_0 + a_1 - 1)\sum_{n=0}^{j-2} f_n^k + \\ &+ 2(a_{-1} + a_0 + a_1 - 1)f_{j-1}^k + 2(a_0 + a_1)f_j^k + 2a_1f_{j+1}^k. \end{aligned}$$

Por tanto:

$$\begin{cases} b_{-1} = -2a_{-2}, & \hat{b}_{-1} = 2a_{-2}; \\ b_0 = 2(a_0 + a_1), & \hat{b}_0 = 2(a_{-2} + a_{-1}); \\ b_1 = 2a_1, & \hat{b}_1 = -2a_1. \end{cases}$$

Si utilizamos el interpolador segmentario lineal de 4 *puntos tenemos que,*

$$(\mathcal{P}_{k-1}^k F^{k-1})_{2j-1} = -\frac{1}{16}F_{j-2}^{k-1} + \frac{9}{16}F_{j-1}^{k-1} + \frac{9}{16}F_j^{k-1} - \frac{1}{16}F_{j+1}^{k-1},$$

así tenemos $a_{-2} = a_1 = -\frac{1}{16}$ *y* $a_{-1} = a_0 = \frac{9}{16}$, *obteniendo* $b_{-1} = -\hat{b}_{-1} = \frac{1}{8}$, $b_0 = \hat{b}_0 = 1$ *y* $b_1 = -\hat{b}_1 = -\frac{1}{8}$, *que es el interpolador lineal centrado utilizando tres celdas celda que vimos en la Ecuación (2.32) de la §2.2.3:*

$$\begin{cases} (\mathcal{P}_{k-1}^k f^{k-1})_{2j-1} = \frac{1}{8}f_{j-1}^{k-1} + f_j^{k-1} - \frac{1}{8}f_{j+1}^{k-1}, \\ (\mathcal{P}_{k-1}^k f^{k-1})_{2j} = -\frac{1}{8}f_{j-1}^{k-1} + f_j^{k-1} + \frac{1}{8}f_{j+1}^{k-1}. \end{cases}$$

Siguiendo el mismo razonamiento y la misma notación si queremos utilizar un interpolador centrado de $5, 7$ *y* 9 *celdas tenemos que*

	$r=4$	$r=6$	$r=8$	$r=10$
b_{-4}				$-2a_{-5}$
b_{-3}			$-2a_{-4}$	$-2(a_{-5}+a_{-4})$
b_{-2}		$-2a_{-3}$	$-2(a_{-4}+a_{-3})$	$-2(a_{-5}+a_{-4}+a_{-3})$
b_{-1}	$-2a_{-2}$	$-2(a_{-3}+a_{-2})$	$-2(a_{-4}+a_{-3}+a_{-2})$	$-2(a_{-5}+a_{-4}+a_{-3}+a_{-2})$
b_0	$2(a_0+a_1)$	$2(a_0+a_1+a_2)$	$2(a_0+a_1+a_2+a_3)$	$2(a_0+a_1+a_2+a_3+a_4)$
b_1	$2a_1$	$2(a_1+a_2)$	$2(a_1+a_2+a_3)$	$2(a_1+a_2+a_3+a_4)$
b_2		$2a_2$	$2(a_2+a_3)$	$2(a_2+a_3+a_4)$
b_3			$2a_3$	$2(a_3+a_4)$
b_4				$2a_4$

Sabiendo que $b_i = -\hat{b}_i$, $-4 \leq i \leq -1$, $1 \leq i \leq 4$; *y* $b_0 = 2 - \hat{b}_0$ *por la relación de consistencia de los operadores.*

Si los filtros obtenidos para valores puntuales son simétricos y suman 1, es decir

$$\sum_{i=-s}^{s-1} a_i = 1, \quad s \in \mathbb{N}^*;$$
$$a_{-s} = a_{s-1}, \quad s \in \mathbb{N},$$

entonces tendríamos que:

	$r=4$	$r=6$	$r=8$	$r=10$
b_{-4}				$2a_4$
b_{-3}			$2a_3$	$2(a_3+a_4)$
b_{-2}		$2a_2$	$2(a_2+a_3)$	$2(a_2+a_3+a_4)$
b_{-1}	$2a_1$	$2(a_1+a_2)$	$2(a_1+a_2+a_3)$	$2(a_1+a_2+a_3+a_4)$
b_0	1	1	1	1
b_1	$2a_1$	$2(a_1+a_2)$	$2(a_1+a_2+a_3)$	$2(a_1+a_2+a_3+a_4)$
b_2		$2a_2$	$2(a_2+a_3)$	$2(a_2+a_3+a_4)$
b_3			$2a_3$	$2(a_3+a_4)$
b_4				$2a_4$

Con el ejemplo visto anteriormente podemos hallar todos los filtros para medias en celda basándonos en las tablas que aparecen en la §3.4.2.

Segunda alternativa: Multiresolución de núcleo en medias en celda

Sea una celda arbitraria $c_\gamma^k = [x_{\gamma-1}^k, x_\gamma^k]$, entonces aproximamos la media de la función $f(x)$ en esa celda por la media en c_γ^k de un polinomio

$z(x) \in \Pi_1^r(\mathbb{R})$ cuyos coeficientes hallamos resolviendo el siguiente problema:

$$\underset{z(x)\in\Pi_1^r(\mathbb{R})}{\arg\min} \sum_{j=1}^{2^{k-1}} K_\lambda(c_\gamma^k, c_j^{k-1})\big(f_j^{k-1} - 2^{k-1}\int_{c_j^{k-1}} z(x)dx\big)^2, \tag{3.17}$$

donde K_λ es una función peso entre las celdas (no introducimos notación nueva[1]). Definimos el núcleo de dos celdas como:

$$K_\lambda(c_\gamma^k, c_j^{k-1}) = \omega\left(\frac{d(c_\gamma^k, c_j^{k-1})}{\lambda}\right), \tag{3.18}$$

donde d es una pseudodistancia entre celdas. Se pueden utilizar muchas distancias (o pseudo) entre dos conjuntos (p. ej. distancia de Haussdorf), nosotros en nuestro caso definimos la distancia como:

$$d(c_\gamma^k, c_j^{k-1}) = \text{máx}\{\min_{x\in c_j^{k-1}} |m_\gamma^k - x|, \min_{x\in c_\gamma^k} |m_j^{k-1} - x|\}, \tag{3.19}$$

donde m_γ^k es el punto medio de la celda c_γ^k y m_j^{k-1} es el punto medio de la celda c_j^{k-1}. Con esta definición si $c_\gamma^k \subset c_j^{k-1}$ entonces $d(c_\gamma^k, c_j^{k-1}) = 0$. Veamos un esquema de la distancia que obtendríamos en la Figura 3.6.

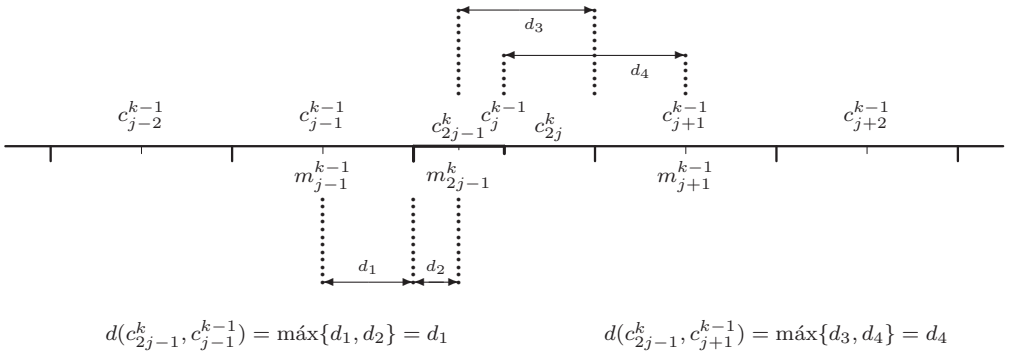


Figura 3.6: *Distancia entre celdas en una dimensión.*

Al igual que hacíamos en el contexto de valores puntuales, §3.3.4, utilizando una notación similar, vemos la linealidad del operador para

[1]Para no recargar el trabajo con excesiva notación denotaremos del mismo modo el peso K_λ entre dos valores puntuales y el peso K_λ definido en la Ecuación (3.18). Para diferenciarlos tan solo debemos darnos cuenta sobre quien están actuando, sobre valores puntuales o sobre celdas.

$p = 2$. Si $z(x) \in \Pi_1^r(\mathbb{R})$ entonces $z(x) = \sum_{i=0}^r \beta_i x^i$, sea $\beta = (\beta_0, \ldots, \beta_r)^T$ y $f^{k-1} = (f_1^{k-1}, \ldots, f_n^{k-1})^T$, sea la matriz $n \times (r+1)$ con $n = 2^{k-1}$

$$\mathbf{X} = \begin{pmatrix} x_1^{k-1} - x_0^{k-1} & \frac{1}{2}(x_1^{k-1})^2 - \frac{1}{2}(x_0^{k-1})^2 & \ldots & \frac{1}{r+1}(x_1^{k-1})^{r+1} - \frac{1}{r+1}(x_0^{k-1})^{r+1} \\ x_2^{k-1} - x_1^{k-1} & \frac{1}{2}(x_2^{k-1})^2 - \frac{1}{2}(x_1^{k-1})^2 & \ldots & \frac{1}{r+1}(x_2^{k-1})^{r+1} - \frac{1}{r+1}(x_1^{k-1})^{r+1} \\ \vdots & \vdots & \ddots & \vdots \\ x_n^{k-1} - x_{n-1}^{k-1} & \frac{1}{2}(x_n^{k-1})^2 - \frac{1}{2}(x_{n-1}^{k-1})^2 & \ldots & \frac{1}{r+1}(x_n^{k-1})^{r+1} - \frac{1}{r+1}(x_{n-1}^{k-1})^{r+1} \end{pmatrix}; \tag{3.20}$$

y la matriz $n \times n$

$$\mathbf{W}_\lambda(c_\gamma^k) = \begin{pmatrix} \omega\big(\frac{d(c_1^{k-1}, c_\gamma^k)}{\lambda}\big) & 0 & \ldots & 0 \\ 0 & \omega\big(\frac{d(c_2^{k-1}, c_\gamma^k)}{\lambda}\big) & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & \omega\big(\frac{d(c_n^{k-1}, c_\gamma^k)}{\lambda}\big) \end{pmatrix}. \tag{3.21}$$

Nota 3.5. *Suponemos, al igual que hicimos en valores puntuales, que los pesos para todas las celdas del muestreo son mayores estrictamente que cero. Si no fuese así reducimos el problema como ya hicimos en la Nota 3.4.*

Entonces podemos ver el problema para $p = 2$ como la solución del siguiente sistema

$$\mathbf{X}^T \mathbf{W}_\lambda(c_\gamma^k) \mathbf{X} \beta = \mathbf{X}^T \mathbf{W}_\lambda(c_\gamma^k) f^{k-1}, \tag{3.22}$$

cuya solución si $|\mathbf{X}^T \mathbf{W}_\lambda(x_\gamma^k)\mathbf{X}| \neq 0$ es

$$\hat{\beta} = (\mathbf{X}^T \mathbf{W}_\lambda(c_\gamma^k)\mathbf{X})^{-1} \mathbf{X}^T \mathbf{W}_\lambda(c_\gamma^k) f^{k-1}.$$

Si definimos para una celda arbitraria $c_j^k = [x_{j-1}^k, x_j^k]$ la función $A(c_j^k) = 2^k(x_j^k - x_{j-1}^k, \frac{1}{2}(x_j^k)^2 - \frac{1}{2}(x_{j-1}^k)^2, \ldots, \frac{1}{r+1}(x_j^k)^{r+1} - \frac{1}{r+1}(x_{j-1}^k)^{r+1})$ entonces tenemos que:

$$\begin{aligned} f_\gamma^k &= A(c_\gamma^k)^T (\mathbf{X}^T \mathbf{W}_\lambda(c_\gamma^k)\mathbf{X})^{-1} \mathbf{X}^T \mathbf{W}_\lambda(c_\gamma^k) f^{k-1} \\ &= \sum_{j=1}^n l_j(c_\gamma^k) f_j^{k-1} \end{aligned} \tag{3.23}$$

donde $l_j(c_\gamma^k) = A(c_\gamma^k)^T (\mathbf{X}^T \mathbf{W}_\lambda(c_\gamma^k)\mathbf{X})^{-1} \mathbf{X}^T \mathbf{W}_\lambda(c_\gamma^k) e_j$, con $e_j = (\delta_{i,j})_{i=1}^n$ que no depende de f^{k-1} y por tanto nuestro **operador es lineal**.

Si definimos así nuestro operador predicción no tendremos garantizada la consistencia. Utilizaremos en este caso la estrategia (AY) presentada en la §2.3.

3.4.4

Multiresolución para medias *hat*

Veamos ahora como diseñar la multiresolución basada en medias *hat*. Para ello utilizamos

$$H(x) = \int_0^x \int_0^y f(z)dzdy, \quad f(x) = \frac{d^2}{dx^2}H(x). \tag{3.24}$$

Sabiendo la biyección existente entre f^k y H^k vista en el capítulo anterior:

$$H_j^k = 4^{-k} \sum_{m=0}^{j-1} \sum_{n=0}^{m} f_n^k, \quad f_j^k = 4^k(H_{j-1}^k - 2H_j^k + H_{j+1}^k), \quad j = 1, \ldots, J_k; \tag{3.25}$$

obtenemos utilizando la misma notación que en la §2.2.2, calculando los coeficientes b_i, $i \in I$, del operador predicción $(\mathcal{P}_{k-1}^k f^{k-1})_{2j-1}$, y recordando que $(\mathcal{P}_{k-1}^k f^{k-1})_{2j} = 2f_j^{k-1} - \frac{1}{2}(f_{2j-1}^k + f_{2j+1}^k)$:

	$r=4$	$r=6$
$\vdots$	$\vdots$	$\vdots$
b_{-5}	$4(3-2a_{-1}-4a_0-6a_1)$	$4(5-2a_{-2}-4a_{-1}-6a_0-8a_1-10a_2)$
b_{-4}	$4(3-2a_{-1}-4a_0-6a_1)$	$4(5-2a_{-2}-4a_{-1}-6a_0-8a_1-10a_2)$
b_{-3}	$4(3-2a_{-1}-4a_0-6a_1)$	$4(5-2a_{-2}-4a_{-1}-6a_0-8a_1-10a_2)$
b_{-2}	$4(3-2a_{-1}-4a_0-6a_1)$	$4(3-2a_{-1}-4a_0-6a_1-8a_2)$
b_{-1}	$4(1-2a_0-4a_1)$	$4(1-2a_0-4a_1-6a_2)$
b_0	$-8a_1$	$4(-2a_1-4a_2)$
b_1		$-8a_1$
b_2		
	$r=8$	$r=10$
$\vdots$	$\vdots$	$\vdots$
b_{-5}	$4(9-\sum_{t=1}^{8} 2ta_{t-5})$	$4(9-\sum_{t=1}^{9} 2ta_{t-5})$
b_{-4}	$4(7-\sum_{t=1}^{7} 2ta_{t-4})$	$4(7-\sum_{t=1}^{8} 2ta_{t-4})$
b_{-3}	$4(5-2a_{-2}-4a_{-1}-6a_0-8a_1-10a_2-12a_3)$	$4(5-2a_{-2}-4a_{-1}-6a_0-8a_1-10a_2-12a_3-14a_4)$
b_{-2}	$4(3-2a_{-1}-4a_0-6a_1-8a_2-10a_3)$	$4(3-2a_{-1}-4a_0-6a_1-8a_2-10a_3-12a_4)$
b_{-1}	$4(1-2a_0-4a_1-6a_2-8a_3)$	$4(1-2a_0-4a_1-6a_2-8a_3-10a_4)$
b_0	$4(-2a_1-4a_2-6a_3)$	$4(-2a_1-4a_2-6a_3-8a_4)$
b_1	$4(-2a_2-4a_3)$	$4(-2a_2-4a_3-6a_4)$
b_2	$-8a_3$	$4(-2a_3-4a_4)$
b_3		$-8a_4$

Si los filtros obtenidos para valores puntuales son simétricos y suman 1 como en la sección anterior entonces tendríamos que:

	$r=4$	$r=6$	$r=8$	$r=10$
b_{-4}				$-8a_4$
b_{-3}			$-8a_3$	$4(-2a_3-4a_4)$
b_{-2}		$-8a_2$	$4(-2a_2-4a_3)$	$4(-2a_2-4a_3-6a_4)$
b_{-1}	$-8a_1$	$4(-2a_1-4a_2)$	$4(-2a_1-4a_2-6a_3)$	$4(-2a_1-4a_2-6a_3-8a_4)$
b_0	$-8a_1$	$4(-2a_1-4a_2)$	$4(-2a_1-4a_2-6a_3)$	$4(-2a_1-4a_2-6a_3-8a_4)$
b_1		$-8a_2$	$4(-2a_2-4a_3)$	$4(-2a_2-4a_3-6a_4)$
b_2			$-8a_3$	$4(-2a_3-4a_4)$
b_3				$-8a_4$

3.4.5
Orden y estabilidad del esquema de multiresolución

El orden definido en el capítulo anterior nos indica qué capacidad tiene el esquema de multiresolución de reproducir un polinomio de un determinado grado r. El orden intenta determinar la calidad del operador predicción, sin embargo, tener un orden alto no garantiza esa supuesta calidad dependiendo de la función, señal o imagen que estemos tratando.

En multiresolución utilizando métodos de núcleo calcular el orden va a ser inmediato utilizando las siguientes proposiciones.

Proposición 3.1. *Con la notación utilizada durante todo el capítulo, sea* $b_s(x_\gamma^k) = \sum_{i=1}^n l_i(x_\gamma^k)(x_\gamma^k - x_i^{k-1})^s$, *con* $s \in \mathbb{N}$ *entonces* $b_0(x_\gamma^k) = 1$ *para regresión local polinómica de cualquier grado (incluido constantes) y* $b_s(x_\gamma^k) = 0$ *para todo* $s \in \{1, 2, \ldots, r\}$ *utilizando regresión polinómica local de grado* r.

Proposición 3.2. *Con la notación utilizada durante todo el capítulo, se cumple que*

$$\sum_{i=1}^{n} l_i(x_\gamma^k)(x_i^{k-1})^s = (x_\gamma^k)^s, \tag{3.26}$$

para todo $s \in \{0, 1, 2, \ldots, r\}$ *utilizando regresión polinómica local de grado* r.

Corolario 3.1. *Con la notación utilizada durante todo el capítulo, se cumple que*

$$\sum_{i=1}^{n} l_i(x_\gamma^k)\left(\sum_{t=0}^{s} a_t(x_i^{k-1})^t\right) = \sum_{t=0}^{s} a_t(x_\gamma^k)^t, \tag{3.27}$$

con $a_i \in \mathbb{R}$ $i = 0, \ldots, s$ *para todo* $s \in \{0, 1, 2, \ldots, r\}$ *utilizando regresión polinómica local de grado* r.

Teorema 3.1. *Con la notación utilizada durante todo el capítulo, el orden del esquema de multiresolución utilizando regresión polinómica local de grado r con independencia de la función peso K_λ elegida es $r+1$.*

Demostración Sea p un polinomio de grado s, i.e., $p(x) \in \Pi_1^s(\mathbb{R})$, con $s \in \{0, 1, \ldots, r\}$. Supongamos un mallado en el intervalo $[0,1]$ equidistante $\{x_j^k\}_{j=0}^{J_k}$ y el operador discretización en valores puntuales visto en el capítulo anterior $p_j^k = (\mathcal{D}_k p)_j = p(x_j^k)$. Entonces definimos nuestro operador decimación como:

$$p_j^{k-1} = p_{2j}^k, \quad j = 0, \ldots, J_{k-1} = J_k/2.$$

y definimos nuestro operador predicción utilizando regresión polinómica local de grado r (misma notación utilizada durante el capítulo) como,

$$\begin{cases} (\mathcal{P}_{k-1}^k p^{k-1})_{2j} = p_j^{k-1}, & j = 0, \ldots, J_{k-1}; \\ (\mathcal{P}_{k-1}^k p^{k-1})_{2j-1} = \sum_{i=0}^{J_{k-1}} l_i(x_{2j-1}^k) p_i^{k-1}, & j = 1, \ldots, J_{k-1}. \end{cases}$$

Debido a la definición de operador decimación y por la Ecuación (3.27) del Corolario 3.1 tenemos que

$$(\mathcal{P}_{k-1}^k p^{k-1})_{2j} = p_j^{k-1} = p_{2j}^k = p(x_{2j}^k);$$
$$(\mathcal{P}_{k-1}^k p^{k-1})_{2j-1} = \sum_{i=0}^{J_{k-1}} l_i(x_{2j-1}^k) p_i^{k-1} = \sum_{i=0}^{J_{k-1}} l_i(x_{2j-1}^k) p(x_i^{k-1}) = p(x_{2j-1}^k).$$

■

Nota 3.6. *En el caso de medias en celda tenemos el mismo orden: si utilizamos la primera alternativa gracias a la función $F(x)$, Ecuación (2.28); y si utilizamos la segunda alternativa usamos la estrategia (AY) que como se prueba en el Lema 2.1 de la §2.3 no altera el orden del esquema. En el caso de medias* hat *es gracias a la función $H(x)$, Ecuación (2.35).*

Por tanto, si utilizamos regresión local polinómica de grado r tendremos asegurada la reproducción exacta del conjunto de polinomios de grado menor o igual a r con independencia de la función peso que utilicemos K_λ y del ancho de banda que tomemos, λ, es decir podemos

tomar información de gran cantidad de puntos sin aumentar el orden del esquema de multiresolución.

Probado el orden de aproximación, veamos la estabilidad del esquema de multiresolución (Definición 2.3 de la §2.2.1). Ésta nos marca si tenemos (o no) un control sobre pequeñas perturbaciones que hagamos sobre la reordenación de datos que vamos efectuando durante la multiresolución. Esto es de gran utilidad en el contexto de compresión de imágenes digitales ya que al aplicar el algoritmo de multiresolución obtenemos datos reales que tenemos que cuantizar, es decir, dar un formato más adecuado para su almacenamiento. Veámoslo por medio del siguiente lema.

Lema 3.1. *Sea $(V^N, ||\cdot||)$ un espacio de dimensión finita normado y sea $(\mathcal{D}_k^{k-1}, \mathcal{P}_{k-1}^k)_k$ un conjunto de operadores decimación y predicción. Si el operador predicción es lineal para todo k entonces el esquema de multiresolución es estable con respecto a la norma $||\cdot||$.*

Demostración Por la linealidad de los operadores decimación y predicción existen dos matrices, $\mathbf{B}_k^{k-1} \in \mathbb{R}^{J_{k-1}\times J_k}$ y $\mathbf{A}_{k-1}^k \in \mathbb{R}^{J_k \times J_{k-1}}$ tales que:

$$\begin{aligned}\mathcal{D}_k^{k-1} f^k &= \mathbf{B}_k^{k-1} f^k, \quad \forall\, f^k \in V^k,\\ \mathcal{P}_{k-1}^k f^{k-1} &= \mathbf{A}_{k-1}^k f^{k-1}, \quad \forall\, f^{k-1} \in V^{k-1}.\end{aligned}$$

Sea $j_0 \in \mathbb{N}$ y $(f^{j_0}, \hat{f}^{j_0})$ entonces tenemos que para cualquier $k \in \{1, \ldots, j_0\}$,

$$||f^{k-1} - \hat{f}^{k-1}|| = ||\mathcal{D}_k^{k-1} f^k - \mathcal{D}_k^{k-1}\hat{f}^k|| = ||\mathcal{D}_k^{k-1}(f^k - \hat{f}^k)|| \leq ||\mathbf{B}_k^{k-1}||||f^k - \hat{f}^k||$$

por tanto si llamamos $C_1 = (\max_{k=1,\ldots,N} ||\mathbf{B}_k^{k-1}||)^N$ se cumple que,

$$||f^0 - \hat{f}^0|| \leq C_1 ||f^{j_0} - \hat{f}^{j_0}||$$

Sea $k \in \{1, \ldots, j_0\}$ entonces

$$\begin{aligned}||d^k - \hat{d}^k|| &= ||(f^k - \mathcal{P}_{k-1}^k f^{k-1}) - (\hat{f}^k - \mathcal{P}_{k-1}^k \hat{f}^{k-1})||\\ &\leq ||f^k - \hat{f}^k|| + ||\mathcal{P}_{k-1}^k f^{k-1} - \mathcal{P}_{k-1}^k \hat{f}^{k-1}||\\ &= ||f^k - \hat{f}^k|| + ||\mathcal{P}_{k-1}^k (f^{k-1} - \hat{f}^{k-1})||\\ &\leq ||f^k - \hat{f}^k|| + ||\mathbf{A}_{k-1}^k||||f^{k-1} - \hat{f}^{k-1}||\\ &\leq (1 + ||\mathbf{A}_{k-1}^k||||\mathbf{B}_k^{k-1}||)||f^k - \hat{f}^k||\end{aligned}$$

llamamos $C_2 = (1 + \max_{k=1,\ldots,N} ||\mathbf{A}_{k-1}^k||||\mathbf{B}_k^{k-1}||)$ y $C = C_1 C_2$ entonces

$$||f^0 - \hat{f}^0|| \leq C||f^{j_0} - \hat{f}^{j_0}||,$$
$$||d^k - \hat{d}^k|| \leq C||f^{j_0} - \hat{f}^{j_0}||, \quad \forall\, 1 \leq k \leq j_0.$$

Así pues queda probada que el algoritmo de descomposición es estable con respecto a la norma $||\cdot||$. Veamos la estabilidad del algoritmo reconstrucción. Sea $j > 0$ entonces

$$\begin{aligned} ||f^j - \hat{f}^j|| &= ||(\mathcal{P}_{j-1}^j f^{j-1} + d^j) - (\mathcal{P}_{j-1}^j \hat{f}^{j-1} + \hat{d}^j)|| \\ &= ||\mathbf{A}_{j-1}^j (f^{j-1} - \hat{f}^{j-1})|| + ||d^j - \hat{d}^j|| \\ &\leq ||\mathbf{A}_{j-1}^j||||f^{j-1} - \hat{f}^{j-1}|| + ||d^j - \hat{d}^j||. \end{aligned}$$

Tomamos $C = (1 + \max_{k=1,\ldots,N} ||\mathbf{A}_{k-1}^k||)$ y queda probado el lema. ■

Corolario 3.2. *El esquema de multiresolución utilizando regresión polinómica local es estable en valores puntuales.*

Corolario 3.3. *En el esquema de multiresolución utilizando regresión polinómica local en el contexto de medias en celda tenemos control del error utilizando la estrategia (AY).*

Demostración El Lema 2.3 asegura que con la estrategia (AY) tenemos control del error si el operador predicción es lineal. ■

3.4.6 Problemas en la frontera. Posible pérdida de orden

Podemos tener una pérdida de orden en los puntos cercanos a la frontera. Para explicarla veamos la Figura 3.7. Sea k un nivel en la multiresolución, en este caso particular sería $k = 4$, i.e., 17 puntos separados a una distancia $h_4 = 2^{-4}$. Tomamos una función de peso cualquiera (en este caso particular `bisq` notación tomada de [70]) y un determinado λ (en nuestro caso $\lambda = 0{,}22$). Trabajamos en valores puntuales, $x_{2j}^k = x_j^{k-1}$, y debemos hallar los valores de la función en los puntos x_{2j-1}^k a partir del

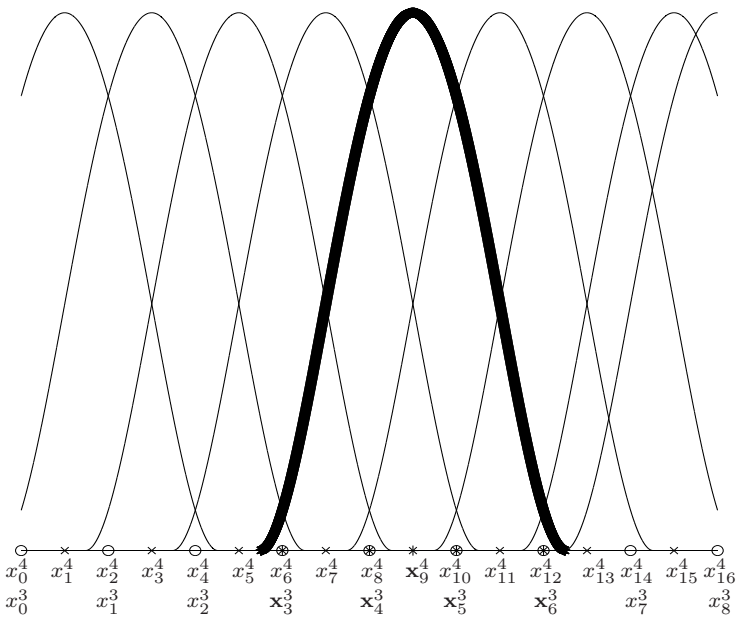


Figura 3.7: *Ejemplo de la función peso en un punto interior.*

nivel $k-1$. Hemos marcado en **negrita** uno de los puntos donde debemos aproximar la función, $\mathbf{x}_9^4$, como podemos ver la apertura de la banda nos proporciona 4 valores para aproximar (dos izquierda y dos derecha):

Sin embargo en los puntos frontera (marcados en **negrita** en la Figura 3.8) no tenemos necesariamente la misma cantidad de puntos para aproximar, así en el ejemplo visto anteriormente de una malla igualmente espaciada en los elementos de la frontera, $\mathbf{x}_1^4$ y $\mathbf{x}_{15}^4$ tenemos 3, $\mathbf{x}_0^3$, $\mathbf{x}_1^3$ y $\mathbf{x}_2^3$; y 2, $\mathbf{x}_7^3$ y $\mathbf{x}_8^3$ puntos para aproximar. Esto puede provocar una pérdida del orden y una mala aproximación en la frontera.

Para solucionar este problema extenderemos el ancho de banda en los puntos fronteras hasta que éste cubra el número de puntos suficientes para poder aproximar con un polinomio del grado que fijamos para hacer el esquema de multiresolución. En la Figura 3.8 si hemos elegido grado 3, entonces necesitamos 4 puntos, así extenderemos el ancho de banda hasta cubrir los puntos $x_0^3, x_1^3, x_2^3, x_3^3$ en la frontera izquierda y hasta cubrir $x_5^3, x_6^3, x_7^3, x_8^3$ en la frontera derecha.

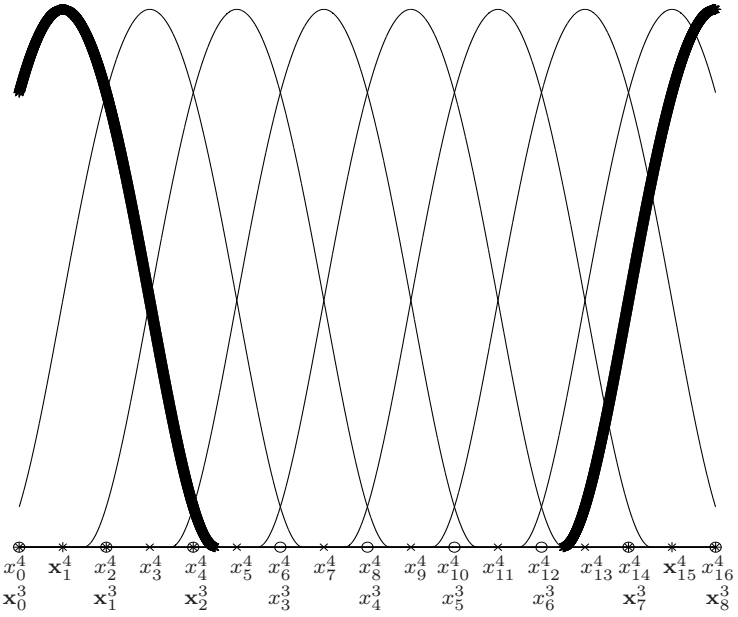


Figura 3.8: *Ejemplo de la función peso en los puntos frontera.*

3.5

Problemas en discontinuidades. Solución: suavizando a cada lado

En muchas ocasiones encontramos ejemplos donde únicamente tenemos suavidad a uno de los lados del punto que queremos aproximar, x_γ^k. Por ejemplo en una discontinuidad de salto de una determinada curva continua a trozos.

Este problema ha sido de gran importancia en los últimos años. Bajo el contexto de métodos núcleo se han desarrollado dos tipos de algoritmos. En primer lugar algoritmos que preservan las discontinuidades sin identificar formalmente los puntos de discontinuidad. El uso de la mediana es un ejemplo simple de este tipo de algoritmo. Métodos más sofisticados son los presentados por McDonald y Owen en [74] o Gijbels et al. en [47].

La segunda clase de algoritmos están basados en dos etapas. En

primer lugar se estima los puntos de la discontinuidad. En segundo lugar se utiliza regresión local entre estos puntos hallados. Podemos ver en la §6.3 de [70] y las referencias que allí hay algunos ejemplos de estos algoritmos.

Para identificar los puntos de discontinuidad se utilizan suavizadores a cada lado. Específicamente, sea x_γ^k un punto a aproximar, definimos la función peso a izquierda y derecha como:

$$K_\lambda^-(x_\gamma^k, x) = \begin{cases} \omega\left(\frac{x_\gamma^k - x}{\lambda}\right), & x < x_\gamma^k; \\ 0, & x \geq x_\gamma^k, \end{cases} \qquad K_\lambda^+(x_\gamma^k, x) = \begin{cases} \omega\left(\frac{x_\gamma^k - x}{\lambda}\right), & x \geq x_\gamma^k; \\ 0, & x < x_\gamma^k, \end{cases}$$

donde la función $\omega(x) \geq 0$ es una función peso elegida. Notar que los pesos $K_\lambda^-(x_\gamma^k, x)$ dan valores distintos de cero solo a aquellos valores $x < x_\gamma^k$, y los pesos $K_\lambda^+(x_\gamma^k, x)$ a los puntos $x \geq x_\gamma^k$. Utilizando estas funciones peso calculamos los polinomios locales obteniendo así los estimadores $\hat{z}_-(x)$ y $\hat{z}_+(x)$ respectivamente.

Los valores $\hat{z}_-(x_\gamma^k)$ y $\hat{z}_+(x_\gamma^k)$ pueden interpretarse como una estimación de los límites a izquierda y a derecha de $z(x)$ cuando $x \to x_\gamma^k$. Si $x_\gamma^k = \tau$ es un punto de discontinuidad entonces estos límites son diferentes, y

$$\Delta(\tau) = \hat{z}_-(\tau) - \hat{z}_+(\tau)$$

es una estimación del tamaño del salto. Cuando x_γ^k esté cerca de una discontinuidad τ la función $\Delta(\tau)$ tendrá un máximo, y estará cercano a 0 cuando no esté cerca de una discontinuidad. Así pues una aproximación de τ se obtiene al maximizar:

$$\hat{\tau} = \arg\min_i \Delta(x_i^k)^2.$$

Después de estimar estos puntos, obtenemos una curva que aproxime a los datos en los diferentes trozos obtenidos aplicando regresión local a cada segmento de datos.

Ejemplo 3.6. *Tomaremos las funciones*

$$f_1(x) = \begin{cases} sin(2\pi x), & x \leq \frac{1}{3}; \\ -sin(2\pi x), & x > \frac{1}{3}; \end{cases} \qquad f_2(x) = \begin{cases} sin(2\pi x), & x \leq \frac{1}{3}; \\ 2 - sin(2\pi x), & x > \frac{1}{3}; \end{cases}$$

$$f_3(x) = \begin{cases} 1, & x \leq \frac{1}{3}; \\ 0, & x > \frac{1}{3}; \end{cases} \qquad f_4(x) = \begin{cases} 100, & x \leq 0{,}375; \\ 240, & 0{,}375 < x < 0{,}79; \\ 100, & x \geq 0{,}79; \end{cases}$$

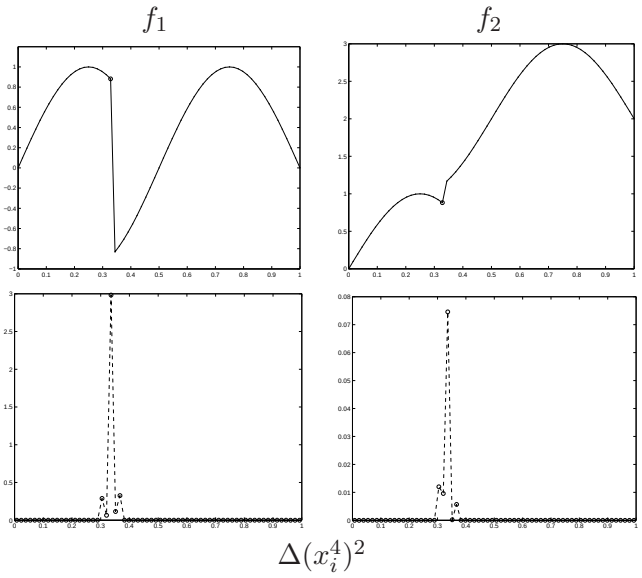


Figura 3.9: *Valores de la función $\Delta(x_i^k)^2$ para las funciones $f_1(x)$ y $f_2(x)$ En la figura de las funciones marcamos con $\circ$ los valores obtenidos como discontinuidades.*

discretizadas en el contexto de valores puntuales con 65 puntos equiespaciados en el intervalo cerrado $[0,1]$. Para calcular $\hat{z}_-(x)$ y $\hat{z}_+(x)$ utilizaremos como núcleo `epan`*, ancho de banda $\lambda = 0{,}05$ y grado del polinomio 1.*

Por tanto como hemos visto en el ejemplo anterior la detección de un salto es clara si las discontinuidades de la función están suficientemente alejadas. Sabemos que alrededor de la discontinuidad la aproximación a un lado (p. ej. derecha) y al otro (izquierda) no es la misma, así determinamos el máximo y eliminamos su entorno (el radio del entorno depende del valor λ que tomamos, en nuestro caso particular tomamos 3 puntos a derecha y 3 a izquierda) para buscar el siguiente máximo relativo. Como en todos los descriptores de singularidades debemos definir *¿qué quiere decir para nosotros singularidad?*, es decir, una cota mínima que nos indique que si un punto supera esta cota entonces es una discontinuidad y si no la supera entonces no será discontinuidad. En los ejemplos anteriores hemos tomado como cota $\nu = 0{,}03$. Así solo podrán ser marcados como máximos aquellos valores que superen esta cota. Veamos un ejemplo de una función con una discontinuidad aislada y una función con ruido que dan sentido a este comentario.

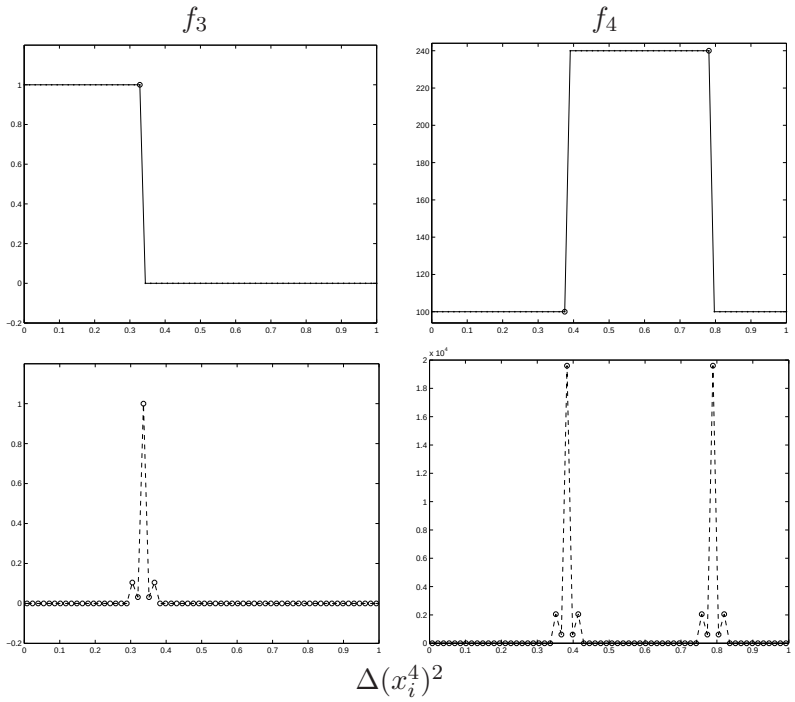


Figura 3.10: *Valores de la función $\Delta(x_i^k)^2$ para las funciones $f_3(x)$ y $f_4(x)$. En la figura de las funciones marcamos con $\circ$ los valores obtenidos como discontinuidades.*

Ejemplo 3.7. *En esta caso tomamos dos funciones, la función f_2 vista en el Ejemplo 3.6 y una función alterada por un ruido gaussiano. Así definimos*

$$f_5(x) = \begin{cases} 3+\epsilon, & x \leq \frac{1}{3}; \\ 7, & x > \frac{1}{3}; \end{cases}$$

donde ϵ es el ruido gaussiano. Marcamos con $\circ$ aquellos puntos que detecta como discontinuidad dependiendo del valor ν dado.

Como podemos ver en el Ejemplo 3.7 al tener una discontinuidad de salto pequeña tenemos que introducir una cota pequeña para que detecte la singularidad. En funciones con cierto nivel de ruido la cota debe ser mayor ya que si no es así marca puntos de ruido como posibles discontinuidades.

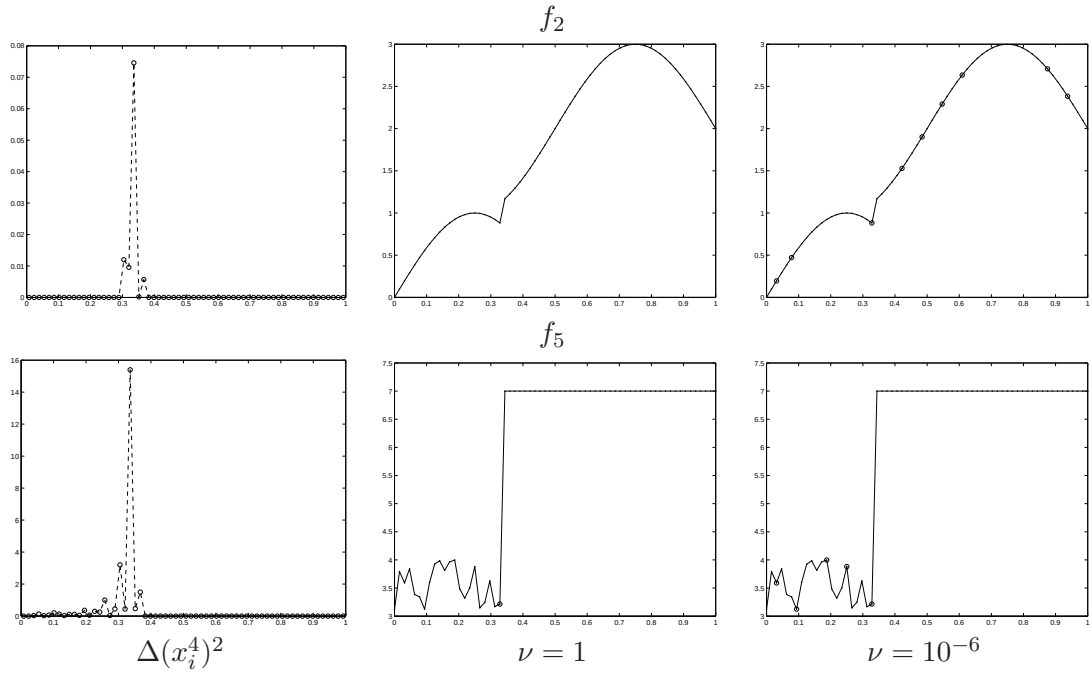


Figura 3.11: *Discontinuidades de las funciones f_2 y f_5 detectadas para distintos valores ν.*

3.6

Cambio en la función de pérdida: norma ℓ^p

El problema (3.3) que planteamos al comienzo del capítulo tenía la forma

$$\hat{z}(x) = \underset{z(x)\in\Pi_1^r(\mathbb{R})}{\arg\min} \sum_{j=1}^{n} K_\lambda(x_\gamma^k, x_j^{k-1}) L(f_j^{k-1}, z(x_j^{k-1}))$$

$$(\hat{\beta}_0, \ldots, \hat{\beta}_r) = \underset{\beta_i\in\mathbb{R}, i=0,\ldots,r}{\arg\min} \sum_{j=1}^{n} K_\lambda(x_\gamma^k, x_j^{k-1}) L(f_j^{k-1}, \sum_{i=0}^{r} \beta_i (x_j^{k-1})^i).$$

Supongamos (con la notación utilizada en la §3.3.4) que la elección de nuestra función de pérdida es la norma ℓ^p, $1 \leq p < +\infty$ entonces obtendríamos el problema

$$\underset{\beta\in\mathbb{R}^{r+1}}{\arg\min} ||\mathbf{W}_\lambda^{1/p}(x_\gamma^k)\mathbf{X}\beta - \mathbf{W}_\lambda^{1/p}(x_\gamma^k) f^{k-1}||_p, \tag{3.28}$$

donde $\mathbf{W}_\lambda^{1/p}(x_\gamma^k)$ es la matriz cuadrada $n \times n$ cuyos elementos no nulos tan solo aparecen en la diagonal y son $(\mathbf{W}_\lambda^{1/p}(x_\gamma^k))_{j,j} = (K_\lambda(x_\gamma^k, x_j^{k-1}))^{1/p}$, $j = 1, \ldots, n$.

Para $p = 2$ es el problema que hemos tratado durante todo el capítulo. Para $p > 2$ el problema es estrictamente convexo por tanto tiene solución única, lo resolvemos utilizando programación lineal convexa (ver [20, 41]).

El caso $p = 1$ es algo más complejo, para ello vamos a utilizar la técnica descrita por T. Chan y P. Mulet en [25, 26]. Nuestro problema sería:

$$\operatorname*{arg\,mín}_{\beta \in \mathbb{R}^{r+1}} \sum_{j=1}^{n} K_\lambda(x_\gamma^k, x_j^{k-1}) \left| f_j^{k-1} - \mathbf{X}_j \beta \right|$$

donde $\mathbf{X}_j = (1, x_j^{k-1}, \ldots, (x_j^{k-1})^r)$, es decir, la j-ésima fila de la matriz $\mathbf{X}$. Como no es diferenciable perturbamos ligeramente la norma ℓ^1 como

$$\operatorname*{arg\,mín}_{\beta \in \mathbb{R}^{r+1}} \sum_{j=1}^{n} K_\lambda(x_\gamma^k, x_j^{k-1}) \sqrt{(f_j^{k-1} - \mathbf{X}_j \beta)^2 + \rho}.$$

siendo ρ un parámetro pequeño positivo. En primer lugar este problema tiene solución pero no es única por tanto añadimos un término que regularice el problema, así:

$$\operatorname*{arg\,mín}_{\beta \in \mathbb{R}^{r+1}} \sum_{j=1}^{n} K_\lambda(x_\gamma^k, x_j^{k-1}) \sqrt{(f_j^{k-1} - \mathbf{X}_j \beta)^2 + \rho} + \frac{\epsilon}{2} ||\beta||_2^2 \tag{3.29}$$

con ϵ un término pequeño. Con este cambio el problema por resultados clásicos en análisis convexo (ver p. ej. [41]) tiene solución única ya que es un operador dos veces continuamente diferenciable, estrictamente convexo, coercivo y acotado inferiormente (ver [10, 26]).

El problema es estable en el sentido de que obtenemos la solución del problema como límite de soluciones del problema perturbado cuando $\rho \to 0$ (ver [1]). Denotaremos $\sqrt{(f_j^{k-1} - \mathbf{X}_j \beta)^2 + \rho}$ como $\left| f_j^{k-1} - \mathbf{X}_j \beta \right|_\rho$. La condición para resolver el problema (3.29) es:

$$\sum_{j=1}^{n} K_\lambda(x_\gamma^k, x_j^{k-1}) \mathbf{X}_j^T \left(\frac{\mathbf{X}_j \beta - f_j^{k-1}}{|f_j^{k-1} - \mathbf{X}_j \beta|_\rho} \right) + \epsilon \beta = 0. \tag{3.30}$$

Vogel y Oman en [91] proponen la siguiente iteración de punto fijo para linealizar la ecuación anterior (3.30):

$$\sum_{j=1}^{n} K_\lambda(x_\gamma^k, x_j^{k-1})\mathbf{X}_j^T\left(\frac{\mathbf{X}_j\beta^{m+1} - f_j^{k-1}}{|f_j^{k-1} - \mathbf{X}_j\beta^m|_\rho}\right) + \epsilon\beta^{m+1} = 0, \tag{3.31}$$

dado β^m hallamos β^{m+1}. En [26] se estudia la convergencia de este método. Veamos la forma matricial de este método. Si llamamos a las matrices $n \times n$

$$\mathbf{D}(\beta^m) = \begin{pmatrix} |f_1^{k-1} - \mathbf{X}_1\beta^m|_\rho^{-1} & 0 & \dots & 0 \\ 0 & |f_2^{k-1} - \mathbf{X}_2\beta^m|_\rho^{-1} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & |f_n^{k-1} - \mathbf{X}_n\beta^m|_\rho^{-1} \end{pmatrix}$$

e $\mathbf{I}_n$ igual a la matriz identidad entonces el sistema a resolver es:

$$(\mathbf{X}^T\mathbf{D}(\beta^m)\mathbf{W}_\lambda(x_\gamma^k)\mathbf{X} + \epsilon\mathbf{I}_n)\beta^{m+1} = \mathbf{X}^T\mathbf{D}(\beta^m)\mathbf{W}_\lambda(x_\gamma^k)f^{k-1},$$

cuya solución si $|\mathbf{X}^T\mathbf{D}(\beta^m)\mathbf{W}_\lambda(x_\gamma^k)\mathbf{X} + \epsilon\mathbf{I}_n| \neq 0$ es:

$$\beta^{m+1} = (\mathbf{X}^T\mathbf{D}(\beta^m)\mathbf{W}_\lambda(x_\gamma^k)\mathbf{X} + \epsilon\mathbf{I}_n)^{-1}\mathbf{X}^T\mathbf{D}(\beta^m)\mathbf{W}_\lambda(x_\gamma^k)f^{k-1}. \tag{3.32}$$

En este caso **el operador no es lineal** por tanto no tenemos asegurada la estabilidad. La convergencia del método es muy rápida, sin embargo el esquema de multiresolución es más lento debido a que debemos hallar el valor óptimo en cada punto.

Veamos los casos en que $p > 2$. Para ello, utilizamos la resolución de problemas de optimización convexa (ver capítulo 6 de [20]). Sea $L_p(x, y) = |x - y|^p$, con $2 < p < +\infty$, está función es convexa, por tanto formulamos nuestro problema (3.28) como:

$$\arg\min_{\beta\in\mathbb{R}^{r+1}} \sum_{j=1}^{n} L_p\big((K_\lambda(x_\gamma^k, x_j^{k-1}))^{1/p} f_j^{k-1}, K_\lambda(x_\gamma^k, x_j^{k-1}))^{1/p}\mathbf{X}_j\beta)\big).$$

La solución la hemos hallado utilizando `cvx`, un paquete que resuelve problemas de optimización convexa presentado por M. Grant et al. en [49, 50].

Si tomamos $p = +\infty$ el problema de aproximación de la norma sería

$$\begin{aligned} &\arg\min_{\beta\in\mathbb{R}^{r+1}} ||\mathbf{W}_\lambda(x_\gamma^k)\mathbf{X}\beta - \mathbf{W}_\lambda(x_\gamma^k)f^{k-1}||_\infty = \\ &\arg\min_{\beta\in\mathbb{R}^{r+1}} \max_{j=1,\dots,n} K_\lambda(x_\gamma^k, x_j^{k-1})\big|f_j^{k-1} - \mathbf{X}_j\beta\big|, \end{aligned} \tag{3.33}$$

que se llama problema de aproximación de Chebyshev o problema de aproximación minimax, ya que minimizamos el máximo (en valor absoluto) de los residuos. Este problema se puede ver como un problema de programación lineal (LP):

$$\begin{array}{ll} \text{mín} & t \\ \text{sujeto a} & -t\mathbf{1} \preceq \mathbf{W}_\lambda(x_\gamma^k)(\mathbf{X}\beta - f^{k-1}) \preceq t\mathbf{1}, \end{array}$$

con variables $\beta \in \mathbb{R}^{r+1}$ y $t \in \mathbb{R}$, donde el símbolo $\preceq$ denota la desigualdad vector o desigualdad componente a componente, es decir si $u, v \in \mathbb{R}^n$ entonces $u \preceq v$ si $u_i \leq v_i$ para $i = 1, \ldots, n$ y $\mathbf{1}$ es el vector $n \times 1$ cuyos elementos son todos unos.

En los casos $p > 2$ la penalización del error para diferencias altas es muy grande, lo que provoca una distorsión en la curva de ajuste. Explicaremos esto con más detalle en la §4.2.2.

Del mismo modo que hemos planteado el problema (3.33) podemos plantear el problema de la norma ℓ^1 como LP (alternativamente a la solución ya planteada),

$$\begin{array}{ll} \text{mín} & \mathbf{1}^T Y \\ \text{sujeto a} & -Y \preceq \mathbf{W}_\lambda^{1-p}(x_\gamma^k)(\mathbf{X}\beta - f^{k-1}) \preceq Y, \end{array}$$

con variables $\beta \in \mathbb{R}^{r+1}$ e $Y \in \mathbb{R}^n$.

Ejemplo 3.8. *Tomamos un mallado equiespaciado de* 128 *puntos en el intervalo cerrado* $[0, 1]$ *con* $h = 1/128$*, es decir* $x_j = jh$ *con* $j = 1, \ldots, 128$*, y sea*

$$f(x) = \begin{cases} e^{x^2} + x^3 + 1, & 0 \leq x \leq 0{,}58; \\ e^{x^2} + x^3 - 1, & 0{,}58 \leq x \leq 1. \end{cases} \tag{3.34}$$

Sea $f_j = f(x_j)$*. Hallamos el conjunto de* 256 *puntos* $\{f(y_j)\}_{j=1}^{256}$ *con* $y_j = j\frac{h}{2}$ *con* $j = 1, \ldots, 256$ *utilizando como función núcleo* `epan` *como grado del polinomio que aproxima* $r = 5$*,* $\lambda = 0{,}04$ *y variamos la función* $L_p(x, y) = |x - y|^p$*, con* $p = 1, 2, 3$ *y la norma en* ℓ^∞*. Como podemos ver en la siguiente figura (Fig. 3.12) utilizando norma* ℓ_1 *no tenemos efecto gibbs en la discontinuidad.*

Ejemplo 3.9. *Utilizamos la misma regresión que en el ejemplo anterior pero con una función escalonada, Fig. 3.13.*

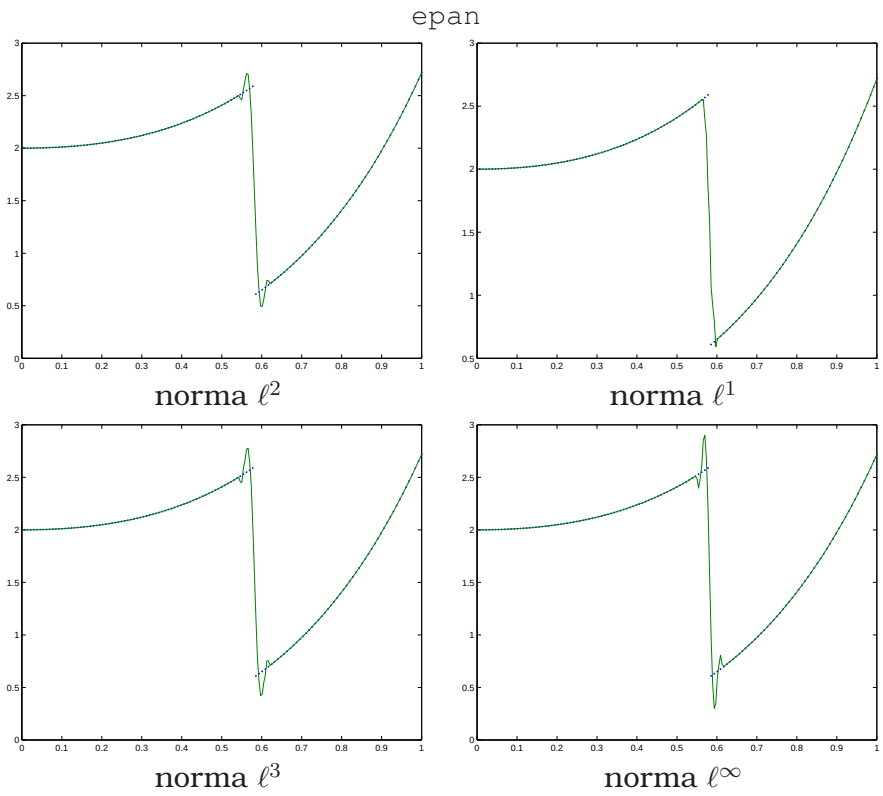


Figura 3.12: *Ejemplo tomando diferentes valores para p con función peso* `epan`*, grado del polinomio $r = 2$ y con $\lambda = 0{,}04$.*

3.7 Experimentos numéricos

Veamos un ejemplo uno dimensional utilizando la discretización basada en valores puntuales (ver §2.2.2). Se podría utilizar medias en celda (ver §3.4.3) pero ya veremos esta discretización en los ejemplos dos dimensionales. Como operador predictor usaremos aquellos basados en métodos de núcleo explicados en este capítulo. Partiremos de un nivel de discretización de $N = 10$ hasta $N_0 = 4$. Sea la función:

$$f_1(x) = \begin{cases} sin(2\pi x), & x \leq \frac{1}{3}; \\ -sin(2\pi x), & x > \frac{1}{3}. \end{cases}$$

Si f^N es un conjunto de datos discretos, denotaremos el error en

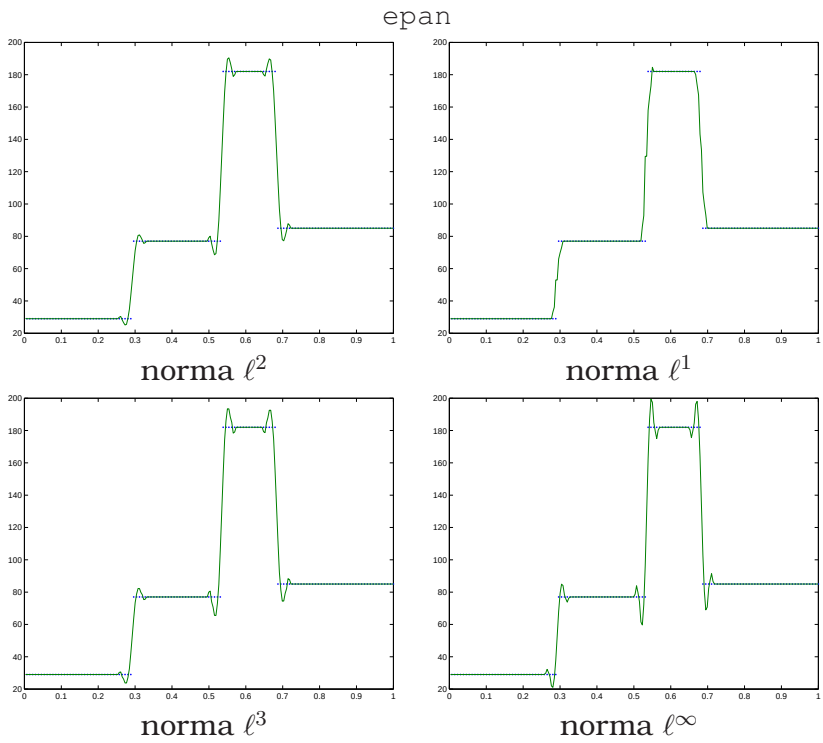


Figura 3.13: *Ejemplo tomando diferentes valores para p con función peso* `epan`*, grado del polinomio $r = 2$ y con $\lambda = 0{,}04$ para una función escalonada.*

distintas normas como

$$E_1 = \frac{1}{J_N}\sum_j |f_j^N - \hat{f}_j^N|, \quad E_2 = \sqrt{\frac{1}{J_N}\sum_j |f_j^N - \hat{f}_j^N|^2}, \quad E_\infty = ||f^N - \hat{f}^N||_\infty.$$

Donde $\hat{f}^N$ es la solución de aplicar el algoritmo de reconstrucción. Marcaremos como NNZ los elementos que almacenamos.

Utilizaremos la estrategia de control del error propuesta por Harten en [53] y desarrollada por Aràndiga et al. en [2] para valores puntuales que consiste en un cambio en el algoritmo de multiresolución (ver p. ej. [2, 3]) basado en hallar los detalles en un determinado nivel k de la multiresolución, truncar aquellos valores menores que un cierto $\epsilon_k = \epsilon$, $\forall k$, es decir:

$$\hat{d}_j^k = \begin{cases} d_j^k, & \text{si } |d_j^k| \geq \varepsilon_k; \\ 0, & \text{si } |d_j^k| < \varepsilon_k. \end{cases}$$

con $j = 0, \ldots, J_k$ y con estos valores hallar $\hat{f}^k$. De manera iterada iríamos calculando la aproximación de la función en todos los niveles hasta llegar a $\hat{f}^N$. En [2, 3] se prueba que aunque utilicemos un operador predicción no lineal (si fuese lineal, como el caso de métodos de núcleo con $p = 2$ ya tenemos estabilidad, ver §3.4.5) tenemos control del error entre $\hat{f}^N$ y f^N.

Utilizaremos el núcleo `gaus` visto en la §3.3.3 y las funciones de pérdida $L_2(x, y)$, $L_1(x, y)$ y $L_\infty(x, y)$ vistas en la sección anterior (§3.6). Una de las principales diferencias que tenemos con el operador clásico definido por interpolación a trozos es que podemos tomar un mayor número de puntos aunque conservemos el grado del polinomio. Tomaremos para cada nivel k un ancho de banda distinto (un número de puntos determinado y fijo), así $\lambda_k = \frac{1}{2}\lambda_{k-1}$, con λ_0 dado. Así pues denotaremos cada método como: ${}^p\texttt{gaus}_m^n$ donde p es 1, 2 o ∞; n es el grado del polinomio que aproxima y m es el número de puntos que utilizamos. El método visto en la §3.5 lo denotaremos como ${}^p\texttt{gaus-s}_m^n$.

Los resultados que obtenemos con este método no son relevantes ya que la singularidad presentada en la función f_1 está aislada. Sin embargo hemos decidido introducir el método en este trabajo como una futura línea de investigación, introducir estas posibilidades ("*one-sided smoothing*") en un esquema de multiresolución podría ser muy ventajoso para tratar funciones con singularidades (ver p. ej. §6.3 de [70] o [47]).

El método clásico de interpolación a trozos lo hemos denotado como PV_m, donde m indica el número de puntos utilizado y por tanto el orden. Este método está también contenido en los métodos de núcleo como hemos visto en este capítulo. En primer lugar vemos en la Tabla 3.1 los resultados para los métodos propuestos.

El método de núcleo `gaus` con 6 puntos con $n = 1$ y $n = 3$ almacena la misma cantidad de detalles que PV_2 y PV_4 respectivamente con funciones de pérdida $L_1(x, y)$ y $L_\infty(x, y)$ produciendo menor error. Se puede ver que PV_6 almacena mayor cantidad de puntos debido a la discontinuidad de salto que tiene la función f_1.

En la Figura 3.14 podemos observar los detalles guardados en cada nivel. Se puede ver que todos los métodos que utilizan un polinomio de grado 1 tienen que guardar todos los detalles en el primer nivel ya que tenemos los puntos para aproximar demasiado alejados unos de otros. Es interesante ver que cuanto mayor es el número de puntos utilizados el efecto *gibbs* en PV aumenta. Sin embargo esto no sucede en los métodos de núcleo por eso mostramos la Tabla 3.2.

El método de núcleo `gaus` en cuanto a valores almacenados se comporta dependiendo del grado y no del número de puntos que tomemos para interpolar. Sin embargo en cuanto a error mejora sustancialmente

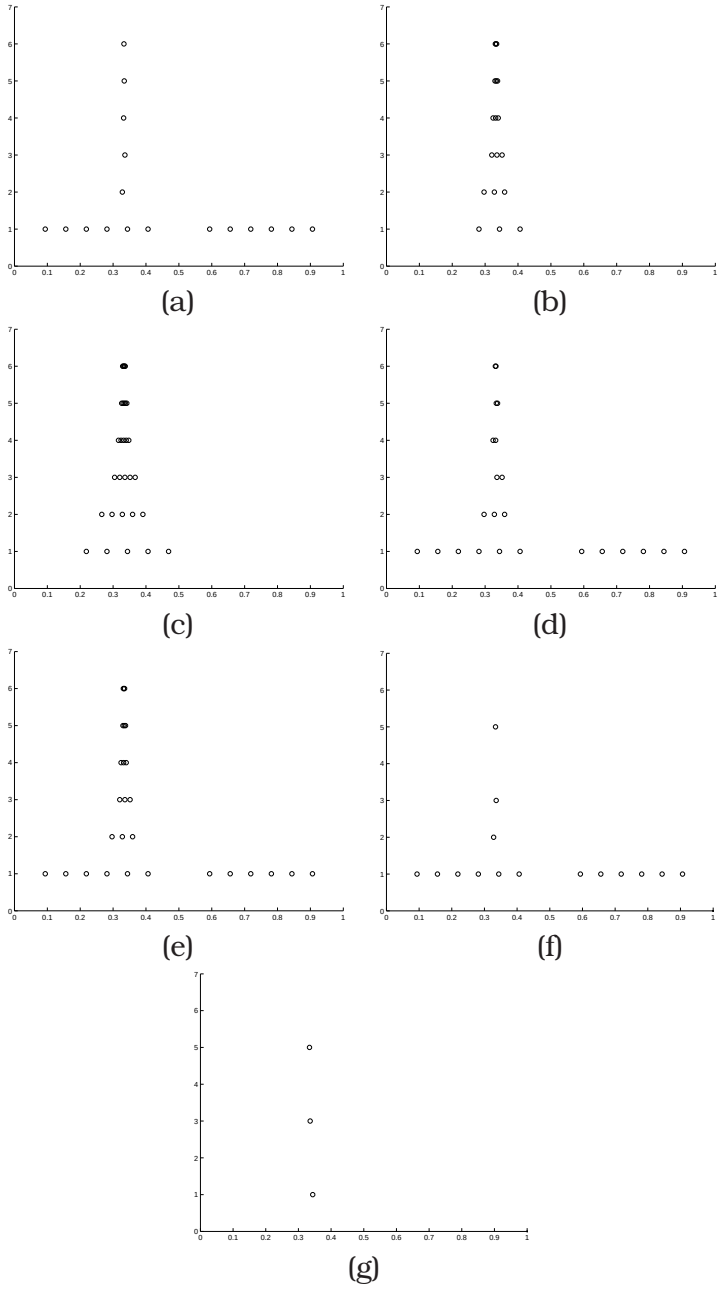


Figura 3.14: *Detalles almacenados, es decir* $|d_j^k| > 10^{-2}$ *utilizando los métodos: (a)* PV_2, $^{1,\infty}\mathtt{gaus}^1_{6,8}$ *(b)* PV_4, $^{2,1,\infty}\mathtt{gaus}^3_{6,8}$ *(c)* PV_6, PV_8, $^{2,1,\infty}\mathtt{gaus}^5_{6,8}$, *(d)* $^{2}\mathtt{gaus}^1_{6}$, *(e)* $^{2}\mathtt{gaus}^1_{8}$, *(f)* $^{2,1,\infty}\mathtt{gaus\text{-}s}^1_{6,8}$, *(g)* $^{2,1,\infty}\mathtt{gaus\text{-}s}^{3,5}_{6,8}$.

	$\varepsilon = 10^{-2}$			
	E_1	E_2	E_∞	NNZ
PV_2	$2{,}43 \cdot 10^{-3}$	$2{,}76 \cdot 10^{-3}$	$3{,}84 \cdot 10^{-3}$	17
PV_4	$4{,}03 \cdot 10^{-4}$	$8{,}83 \cdot 10^{-4}$	$1{,}78 \cdot 10^{-4}$	18
PV_6	$2{,}93 \cdot 10^{-4}$	$8{,}64 \cdot 10^{-4}$	$3{,}00 \cdot 10^{-4}$	30
${}^{2}\texttt{gaus}_6^1$	$2{,}57 \cdot 10^{-3}$	$3{,}00 \cdot 10^{-3}$	$9{,}48 \cdot 10^{-3}$	23
${}^{2}\texttt{gaus}_6^3$	$4{,}35 \cdot 10^{-4}$	$9{,}32 \cdot 10^{-4}$	$3{,}65 \cdot 10^{-3}$	18
${}^{1}\texttt{gaus}_6^1$	$2{,}43 \cdot 10^{-3}$	$2{,}76 \cdot 10^{-3}$	$3{,}84 \cdot 10^{-3}$	17
${}^{1}\texttt{gaus}_6^3$	$1{,}79 \cdot 10^{-4}$	$2{,}44 \cdot 10^{-4}$	$4{,}69 \cdot 10^{-4}$	18
${}^{\infty}\texttt{gaus}_6^1$	$2{,}45 \cdot 10^{-3}$	$2{,}79 \cdot 10^{-3}$	$3{,}84 \cdot 10^{-3}$	17
${}^{\infty}\texttt{gaus}_6^3$	$1{,}79 \cdot 10^{-4}$	$2{,}44 \cdot 10^{-4}$	$4{,}68 \cdot 10^{-4}$	18
${}^{2}\texttt{gaus-s}_6^1$	$2{,}63 \cdot 10^{-3}$	$2{,}97 \cdot 10^{-3}$	$4{,}13 \cdot 10^{-3}$	15
${}^{2}\texttt{gaus-s}_6^3$	$5{,}77 \cdot 10^{-4}$	$1{,}08 \cdot 10^{-3}$	$3{,}65 \cdot 10^{-3}$	3
${}^{1}\texttt{gaus-s}_6^1$	$2{,}44 \cdot 10^{-3}$	$2{,}76 \cdot 10^{-3}$	$3{,}84 \cdot 10^{-3}$	15
${}^{1}\texttt{gaus-s}_6^3$	$2{,}57 \cdot 10^{-4}$	$3{,}92 \cdot 10^{-4}$	$9{,}68 \cdot 10^{-4}$	3
${}^{\infty}\texttt{gaus-s}_6^1$	$2{,}45 \cdot 10^{-3}$	$2{,}79 \cdot 10^{-3}$	$3{,}84 \cdot 10^{-3}$	15
${}^{\infty}\texttt{gaus-s}_6^3$	$2{,}57 \cdot 10^{-4}$	$3{,}91 \cdot 10^{-4}$	$9{,}69 \cdot 10^{-4}$	3

Tabla 3.1: *Errores y detalles almacenados utilizando el método de núcleo* `gaus` *con 6 puntos y* $n = 1, 3, 5$.

los métodos PV siempre y cuando utilicemos como función de pérdida $L_1(x, y)$ y $L_\infty(x, y)$.

En conclusión los métodos de núcleo nos permiten utilizar información de una mayor cantidad de puntos sin elevar demasiado el orden del polinomio que aproxima. Elevar el orden nos puede provocar un fuerte efecto *gibbs*. Así pues nos encontraríamos en "*mitad de dos mundos*": por una parte utilizar un número de puntos elevados (que nos ofrece información) para el operador predictor y por otro no subir demasiado el grado del polinomio que aproxima para que no se produzca un excesivo efecto en las discontinuidades.

3.8 Generalización a dos dimensiones

La extensión formal de la definición de regresión local para un predictor multivariable es directa; requiere una función peso y un polinomio local de varias dimensiones. Fueron McLain en [75] y Stone en [86] quienes

	$\varepsilon = 10^{-2}$			
	E_1	E_2	E_∞	NNZ
PV_2	$2{,}43 \cdot 10^{-3}$	$2{,}76 \cdot 10^{-3}$	$3{,}84 \cdot 10^{-3}$	17
PV_4	$4{,}03 \cdot 10^{-4}$	$8{,}83 \cdot 10^{-4}$	$1{,}78 \cdot 10^{-4}$	18
PV_6	$2{,}93 \cdot 10^{-4}$	$8{,}64 \cdot 10^{-4}$	$3{,}00 \cdot 10^{-4}$	30
PV_8	$8{,}11 \cdot 10^{-4}$	$1{,}50 \cdot 10^{-3}$	$4{,}31 \cdot 10^{-3}$	30
${}^{2}\texttt{gaus}_8^1$	$3{,}57 \cdot 10^{-3}$	$4{,}10 \cdot 10^{-3}$	$6{,}26 \cdot 10^{-3}$	27
${}^{2}\texttt{gaus}_8^3$	$7{,}53 \cdot 10^{-4}$	$1{,}22 \cdot 10^{-3}$	$3{,}79 \cdot 10^{-3}$	18
${}^{2}\texttt{gaus}_8^5$	$1{,}70 \cdot 10^{-5}$	$2{,}43 \cdot 10^{-5}$	$5{,}51 \cdot 10^{-5}$	30
${}^{1}\texttt{gaus}_8^1$	$9{,}52 \cdot 10^{-4}$	$1{,}38 \cdot 10^{-3}$	$3{,}63 \cdot 10^{-3}$	17
${}^{1}\texttt{gaus}_8^3$	$1{,}79 \cdot 10^{-4}$	$2{,}44 \cdot 10^{-4}$	$4{,}69 \cdot 10^{-4}$	18
${}^{1}\texttt{gaus}_8^5$	$9{,}25 \cdot 10^{-6}$	$1{,}73 \cdot 10^{-5}$	$2{,}11 \cdot 10^{-5}$	30
${}^{\infty}\texttt{gaus}_8^1$	$2{,}97 \cdot 10^{-3}$	$3{,}44 \cdot 10^{-3}$	$3{,}89 \cdot 10^{-3}$	17
${}^{\infty}\texttt{gaus}_8^3$	$1{,}79 \cdot 10^{-4}$	$2{,}44 \cdot 10^{-4}$	$4{,}69 \cdot 10^{-4}$	18
${}^{\infty}\texttt{gaus}_8^5$	$1{,}00 \cdot 10^{-5}$	$1{,}73 \cdot 10^{-5}$	$1{,}99 \cdot 10^{-5}$	30
${}^{2}\texttt{gaus-s}_8^1$	$3{,}72 \cdot 10^{-3}$	$4{,}18 \cdot 10^{-3}$	$5{,}83 \cdot 10^{-3}$	15
${}^{2}\texttt{gaus-s}_8^3$	$6{,}01 \cdot 10^{-4}$	$1{,}12 \cdot 10^{-3}$	$3{,}79 \cdot 10^{-3}$	3
${}^{2}\texttt{gaus-s}_8^5$	$1{,}68 \cdot 10^{-5}$	$3{,}46 \cdot 10^{-5}$	$3{,}48 \cdot 10^{-4}$	3
${}^{1}\texttt{gaus-s}_8^1$	$2{,}44 \cdot 10^{-3}$	$2{,}76 \cdot 10^{-3}$	$3{,}84 \cdot 10^{-3}$	15
${}^{1}\texttt{gaus-s}_8^3$	$2{,}57 \cdot 10^{-4}$	$3{,}91 \cdot 10^{-4}$	$9{,}69 \cdot 10^{-4}$	3
${}^{1}\texttt{gaus-s}_8^5$	$1{,}65 \cdot 10^{-5}$	$3{,}46 \cdot 10^{-5}$	$3{,}43 \cdot 10^{-4}$	3
${}^{\infty}\texttt{gaus-s}_8^1$	$2{,}97 \cdot 10^{-3}$	$3{,}45 \cdot 10^{-3}$	$3{,}89 \cdot 10^{-3}$	15
${}^{\infty}\texttt{gaus-s}_8^3$	$2{,}57 \cdot 10^{-4}$	$3{,}91 \cdot 10^{-4}$	$9{,}69 \cdot 10^{-4}$	3
${}^{\infty}\texttt{gaus-s}_8^5$	$1{,}70 \cdot 10^{-5}$	$3{,}60 \cdot 10^{-5}$	$4{,}41 \cdot 10^{-4}$	3

Tabla 3.2: *Errores y detalles almacenados utilizando el método de núcleo* `gaus` *con 8 puntos y* $n = 1, 3, 5$.

introdujeron esta generalización.

Sea $z(x, y) \in \Pi_2^r(\mathbb{R})$, i. e.

$$z(x, y) = \sum_{0 \leq \max\{s,t\} \leq r} \beta_{s,t} x^s y^t, \tag{3.35}$$

sea $\Lambda = (\gamma_1, \gamma_2)$ y un punto arbitrario $X_\Lambda^k = (x_{\gamma_1}^k, y_{\gamma_2}^k)$ entonces definimos nuestra función núcleo como

$$K_\lambda(X_\Lambda^k, X) = \omega\left(\frac{||X_\Lambda^k - X||}{\lambda}\right)$$

donde $\omega(x)$ es una de las funciones peso definidas en la sección §3.3.3. Dependiendo de la norma $||\cdot||$ que tomemos obtendremos una métrica

u otra. Podemos ver en las Figuras 3.15 y 3.16 la comparación entre las normas ℓ^1, la norma euclídea y la norma en ℓ^∞ utilizando diferentes núcleos.

Sea $X_Q^{k-1} = (x_{q_1}^{k-1}, y_{q_2}^{k-1})$ entonces nuestro problema sería:

$$\hat{\beta} = \underset{\beta_{s,t}\in\mathbb{R}, 0\leq \max\{s,t\}\leq r}{\arg\min} \sum_{q_1,q_2=1}^{n} K_\lambda(X_\Lambda^k, X_Q^{k-1})\big(f_{q_1,q_2}^{k-1} - \sum_{0\leq \max\{s,t\}\leq r} \beta_{s,t}(x_{q_1}^{k-1})^s (y_{q_2}^{k-1})^t\big)^2. \tag{3.36}$$

El problema, por tanto, quedaría del mismo modo que en una dimensión. En el contexto de valores puntuales podemos o bien utilizar esta generalización directa de dos dimensiones (como en [7]) o bien utilizar producto tensorial con los filtros obtenidos en una dimensión (ver [2, 3]). El método es estable en el caso dos dimensional ya que es lineal al igual que en el uno dimensional.

Vemos los diferentes puntos que obtendríamos de la escala $k-1$ para construir el operador predictor en los puntos $x_{2i-1,2j}$, $x_{2i-1,2j-1}$ (para el punto $x_{2i,2j-1}$ tenemos los puntos simétricos a los tomados para el punto $x_{2i-1,2j}$) dependiendo del ancho de banda que tomemos y la norma escogida $||\cdot||_p$ con $p = 1, 2, \infty$ en las Figuras 3.17, 3.18, 3.19, 3.20, 3.21, 3.22. Tomamos $\lambda_1 < \ldots < \lambda_6$ como ancho de banda de manera creciente en cuanto a número de puntos utilizados, este valor dependería de la escala tomada.

Sin embargo en el contexto de medias en celda en dos dimensiones debemos definir como ya hicimos en una dimensión la distancia entre celdas (que será una generalización de la ya definida). Veámosla con detalle en la siguiente sección.

3.8.1 Multiresolución en medias en celda en dos dimensiones

Al igual que ocurría en una dimensión vista en la §3.4.3 la generalización a dos dimensiones será análoga, incluso en la definición de la distancia entre dos celdas que utilizaremos para definir los núcleos. Así, sea V^k un conjunto de valores $f_{i,j}^k = 4^k \int_{c_{i,j}^k} f(X)dX$, con $c_{i,j}^k = [(i -$

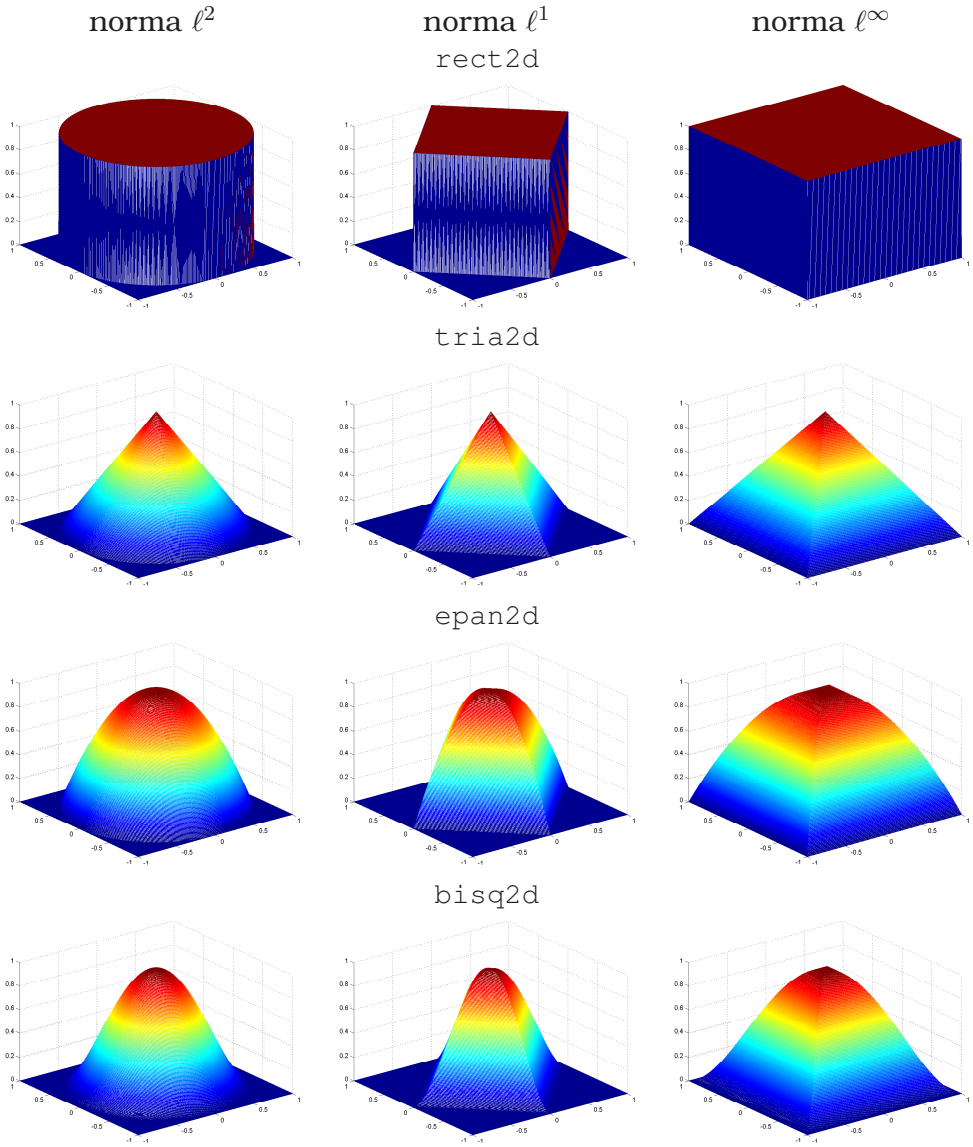


Figura 3.15: *Peso asignado a cada vector* $X \in \mathbb{R}^2$ *utilizando diferentes núcleos:* `rect2d`, `tria2d`, `epan2d` *y* `bisq2d` *con* $\lambda = 1$ *y con norma* ℓ^p, $p = 1, 2, \infty$ *siendo* $X_\Lambda^k = (0, 0)$, *es decir* $K_1((0, 0), X)$.

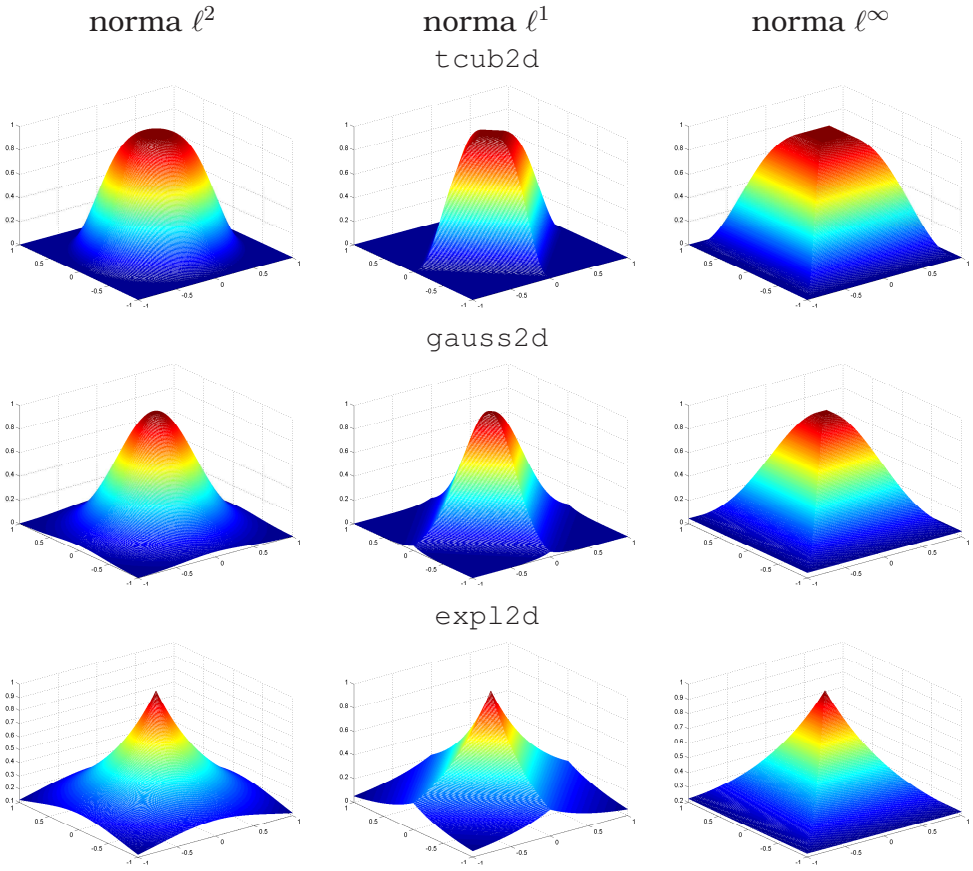


Figura 3.16: *Peso asignado a cada vector $X \in \mathbb{R}^2$ utilizando diferentes núcleos:* `tcub2d`, `gauss2d` *y* `expl2d` *con $\lambda = 1$ y con norma ℓ^p, $p = 1, 2, \infty$ siendo $X^k_\Lambda = (0,0)$, es decir $K_1((0,0), X)$.*

$1)2^{-k}, i2^{-k}] \times [(j-1)2^{-k}, j2^{-k}]$ e $i, j = 1, \ldots, J_k$. Por tanto tenemos que $c^{k-1}_{i,j} = c^k_{2i-1,2j-1} \cup c^k_{2i-1,2j} \cup c^k_{2i,2j-1} \cup c^k_{2i,2j}$, entonces nuestro operador decimación es

$$(\mathcal{D}^{k-1}_k f^k)_{i,j} = \frac{1}{4}(f^k_{2i-1,2j-1} + f^k_{2i-1,2j} + f^k_{2i,2j-1} + f^k_{2i,2j}). \tag{3.37}$$

Veamos cómo definir el operador predicción utilizando el método de núcleo, lo explicitamos solo para la celda $c^k_{2i-1,2j-1}$, de forma análoga sería para $c^k_{2i-1,2j}, c^k_{2i,2j-1}, c^k_{2i,2j}$. Sea $z(x,y) \in \Pi^r_2(\mathbb{R})$ como en la sección anterior y sea c^k_Λ una celda arbitraria entonces definimos nuestra función núcleo

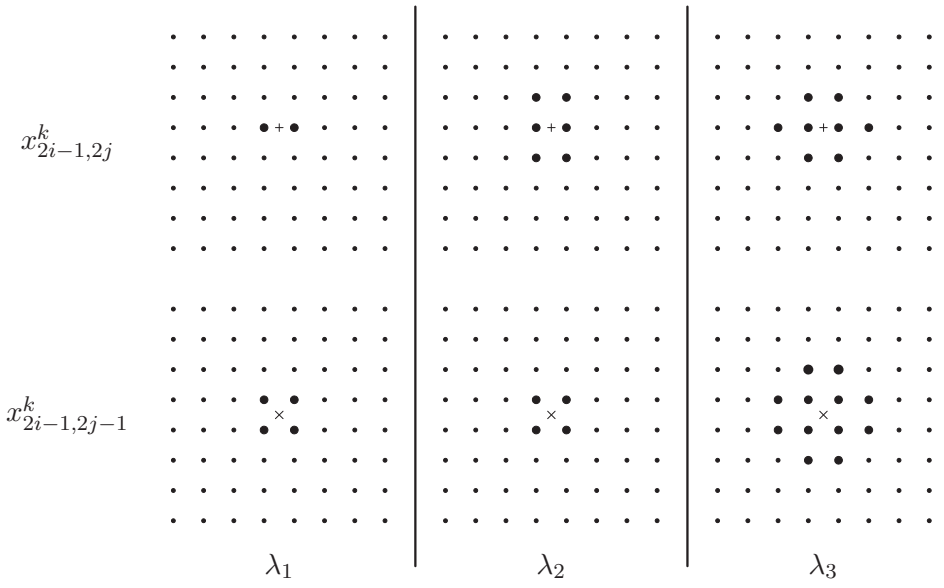


Figura 3.17: *Puntos obtenidos para la predicción, Norma* $||\cdot||_2$.

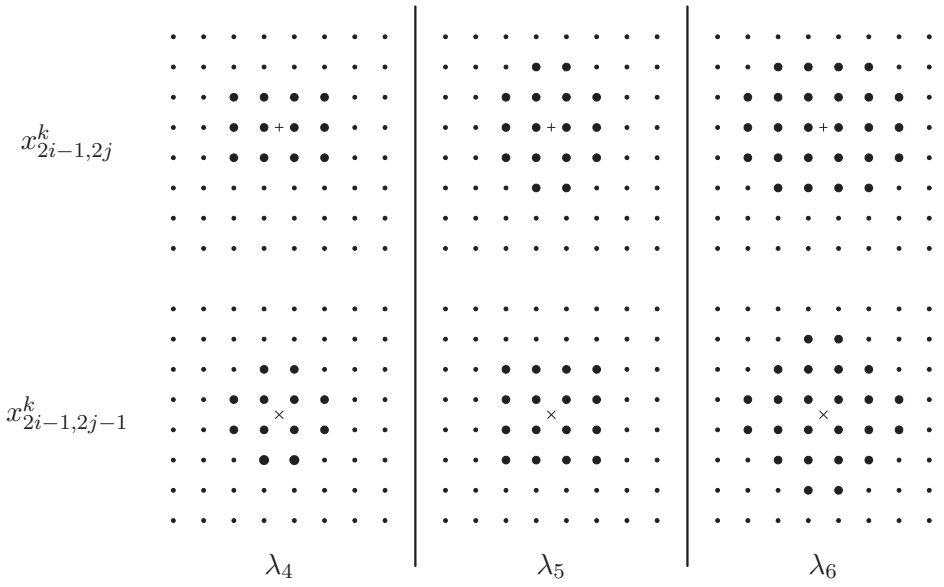


Figura 3.18: *Puntos obtenidos para la predicción, Norma* $||\cdot||_2$.

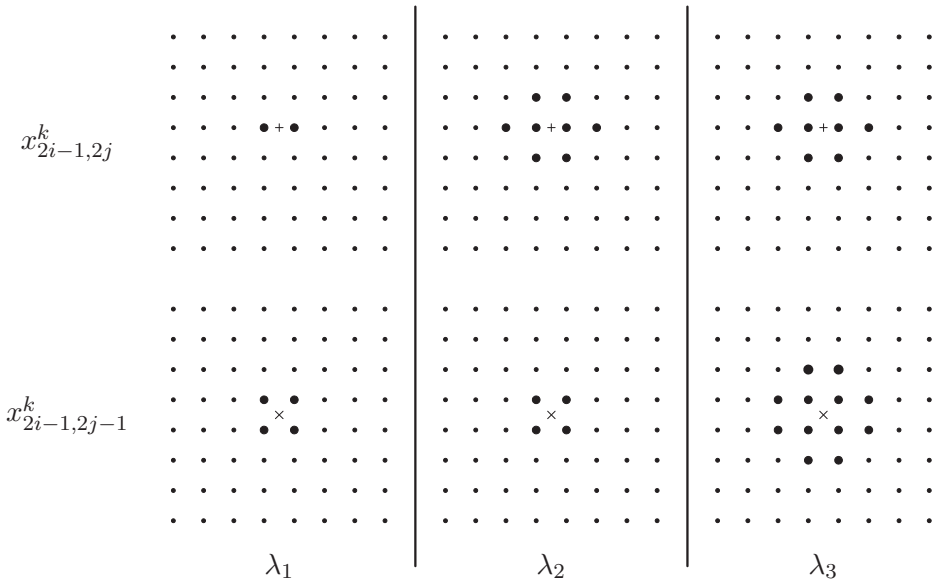


Figura 3.19: *Puntos obtenidos para la predicción, Norma* $||\cdot||_1$.

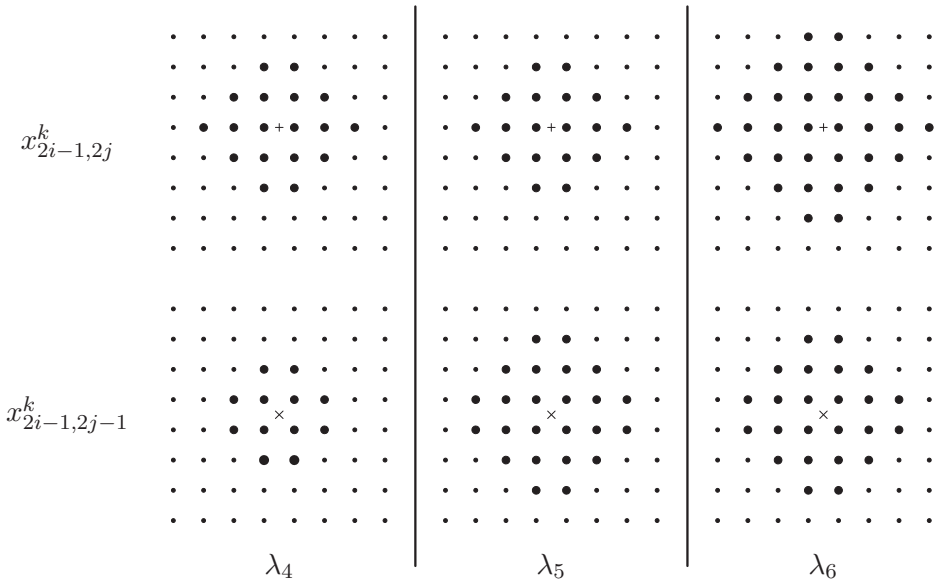


Figura 3.20: *Puntos obtenidos para la predicción, Norma* $||\cdot||_1$.

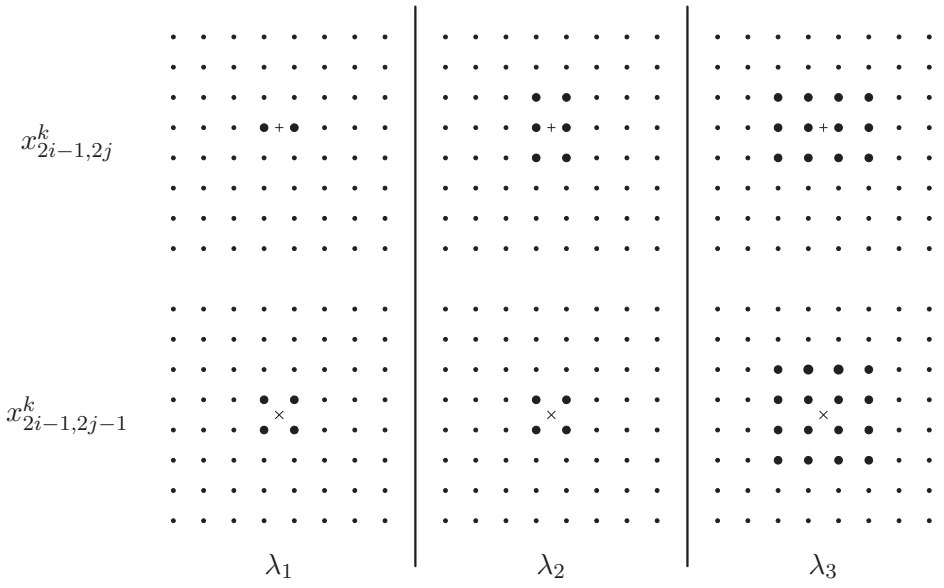


Figura 3.21: *Puntos obtenidos para la predicción, Norma* $||\cdot||_\infty$.

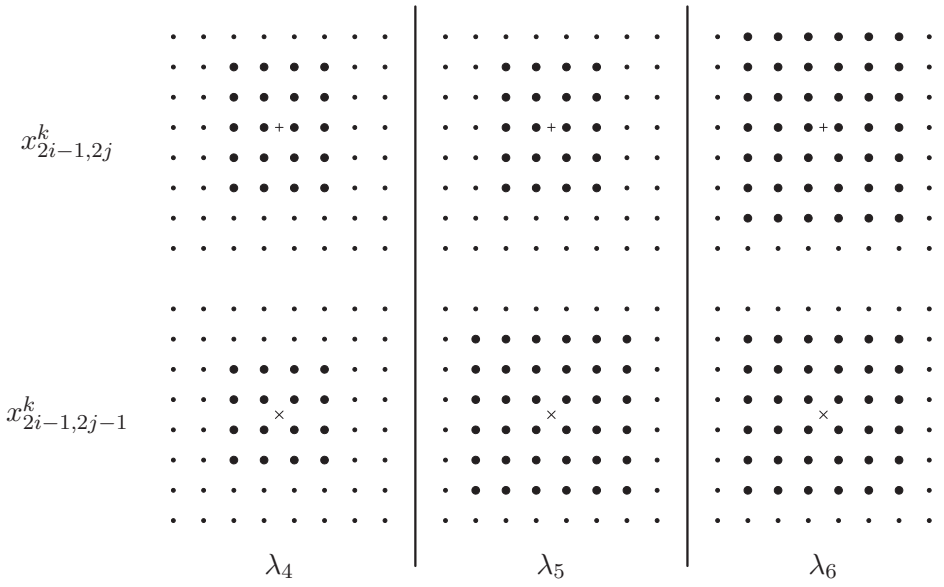


Figura 3.22: *Puntos obtenidos para la predicción, Norma* $||\cdot||_\infty$.

como

$$K_\lambda(c_\Lambda^k, c_{i,j}^{k-1}) = \omega\left(\frac{d(c_\Lambda^k, c_{i,j}^{k-1})}{\lambda}\right),$$

donde d es la distancia entre normas definida como

$$d(c_\Lambda^k, c_{i,j}^{k-1}) = \text{máx}\{\min_{X\in c_\Lambda^k} ||M_{i,j}^{k-1} - X||, \min_{X\in c_{i,j}^{k-1}} ||M_\Lambda^k - X||\},$$

con $||\cdot||$ una norma vectorial; y $M_{i,j}^{k-1}$, y M_Λ^k son los puntos medios de las celdas c_Λ^k y $c_{i,j}^{k-1}$ respectivamente (esto es posible ya que las celdas son cuadradas, en el caso de que las celdas tuviesen una forma no regular podríamos utilizar la distancia de Hausdorff).

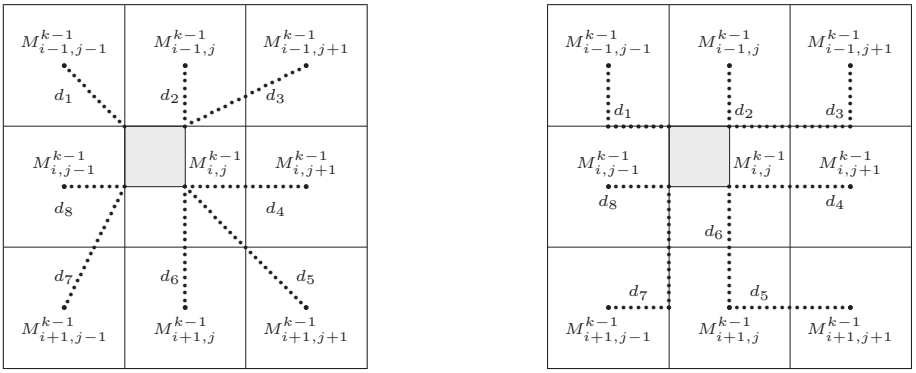


Figura 3.23: *Distancia entre celdas utilizando norma 2 y norma 1, donde* $d_1 = d(c_{2i-1,2j-1}^k, c_{i-1,j-1}^{k-1})$, $d_2 = d(c_{2i-1,2j-1}^k, c_{i-1,j}^{k-1})$, $d_3 = d(c_{2i-1,2j-1}^k, c_{i-1,j+1}^{k-1})$, $d_4 = d(c_{2i-1,2j-1}^k, c_{i,j+1}^{k-1})$, $d_5 = d(c_{2i-1,2j-1}^k, c_{i+1,j+1}^{k-1})$, $d_6 = d(c_{2i-1,2j-1}^k, c_{i+1,j}^{k-1})$, $d_7 = d(c_{2i-1,2j-1}^k, c_{i+1,j-1}^{k-1})$, $d_8 = d(c_{2i-1,2j-1}^k, c_{i,j-1}^{k-1})$ *y* $d_9 = d(c_{2i-1,2j-1}^k, c_{i,j}^{k-1}) = 0$.

Por tanto el problema que tenemos es

$$\begin{aligned}\hat{\beta} = \underset{\beta_{s,t}\in\mathbb{R}, 0\leq \text{máx}\{s,t\}\leq r}{\arg\min} \sum_{i,j=1}^{4^{k-1}} K_\lambda(c_{2i-1,2j-1}^k, c_{i,j}^{k-1})\big(f_{i,j}^{k-1} - \\ -4^{k-1}\int_{c_{i,j}^{k-1}} \sum_{0\leq \text{máx}\{s,t\}\leq r} \beta_{s,t} x^s y^t d(x,y)\big)^2,\end{aligned} \tag{3.38}$$

y el operador predicción será

$$(\mathcal{P}_{k-1}^{k} f^{k-1})_{2i-1,2j-1} = 4^k \int_{c_{2i-1,2j-1}^k} \sum_{0\leq \max\{s,t\}\leq r} \hat{\beta}_{s,t} x^s y^t d(x,y) =$$

$$\sum_{0\leq \max\{s,t\}\leq r} \frac{\hat{\beta}_{s,t}}{2^{(s+t)}} \left(\frac{(2i-1)^{s+1}}{s+1} - \frac{(2i-2)^{s+1}}{s+1} \right) \left(\frac{(2j-1)^{t+1}}{t+1} - \frac{(2j-2)^{t+1}}{t+1} \right).$$

Es fácil ver (como en la §3.3.4) que el operador así definido es lineal, (ver también p. ej. [70]). Al definir un operador predicción en cada celda no tenemos consistencia, así pues utilizaremos la estrategia (AY) descrita en la §2.3.

Veamos en los siguientes esquemas qué celdas del nivel $k-1$ utilizamos si tomamos diferentes anchos de banda $\lambda_1 < \ldots < \lambda_6$ de manera creciente en cuanto a número de celdas utilizadas, Figs. 3.24, 3.25 y 3.26. Si tomamos en el polinomio que aproxima un grado adecuado y un ancho de banda de manera que el sistema que obtenemos es completo y determinado podemos ver que con norma infinito los filtros obtenidos son los mismos que los obtenidos en interpolación polinómica a trozos (ver p. ej. [7, 13, 12]).

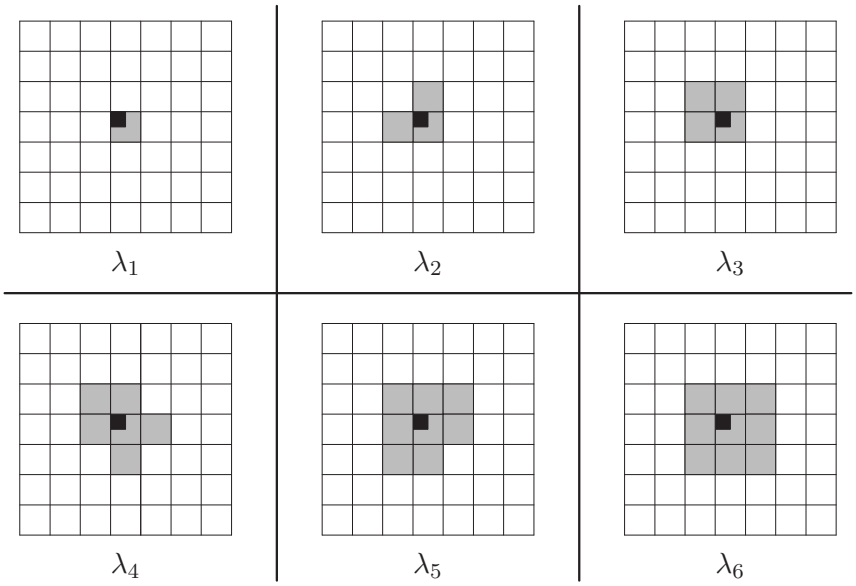


Figura 3.24: *Celdas utilizadas en el nivel $k-1$ para hallar la celda marca, $||\cdot||_2$.*

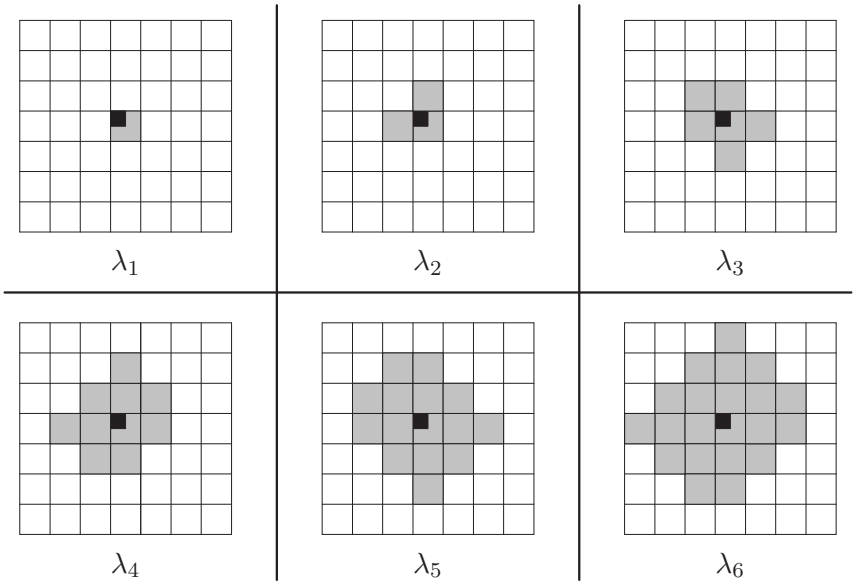


Figura 3.25: *Celdas utilizadas en el nivel* $k-1$ *para hallar la celda marcada,* $||\cdot||_1$.

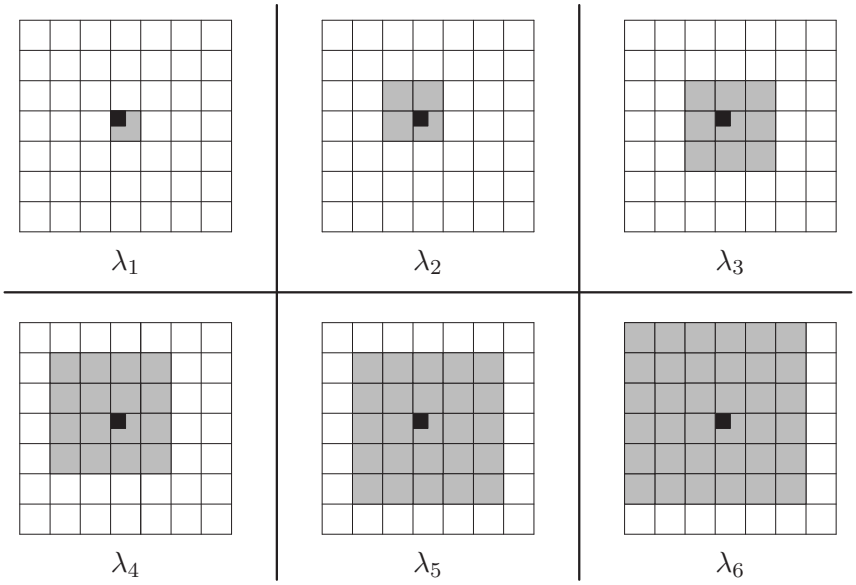


Figura 3.26: *Celdas utilizadas en el nivel* $k-1$ *para hallar la celda marcada,* $||\cdot||_\infty$.

3.9 Experimentos numéricos

En esta sección utilizaremos los algoritmos de multiresolución para el tratamiento de imágenes, en particular lo usaremos para la compresión. Tomaremos tres imágenes: dos imágenes reales, *lena* y *peppers* y una geométrica, *geopa*. Utilizaremos el algoritmo de transformación directa de multiresolución, cuantizaremos los valores que obtengamos y ésos los almacenaremos. Por último reconstruiremos la imagen con estos valores.

En este caso, haremos los experimentos en el contexto de medias en celda (ver §2.2.2) ya que este operador discretización se ajusta mejor a la naturaleza de los datos. Para medir la calidad de la imagen resultante tomaremos el valor PSNR visto en la Ecuación (2.78) del capítulo 2.

Cuanto mayor sea el valor PSNR menor será la distancia entre la imagen original y la imagen reconstruida. En muchas ocasiones a mayores PSNR no necesariamente tenemos mayor calidad visual, por esto también mostraremos una comparación entre las diferentes imágenes obtenidas utilizando cada uno de los métodos.

geopa

Comparamos los siguientes métodos:

- métodos interpolatorios utilizando una ventana de 3×3 puntos, es decir, el producto tensorial del operador:

$$\begin{cases} (\mathcal{P}_{k-1}^{k} f^{k-1})_{2j-1} = \frac{1}{8} f_{j-1}^{k-1} + f_j^{k-1} - \frac{1}{8} f_{j+1}^{k-1}, \\ (\mathcal{P}_{k-1}^{k} f^{k-1})_{2j} = -\frac{1}{8} f_{j-1}^{k-1} + f_j^{k-1} + \frac{1}{8} f_{j+1}^{k-1}, \end{cases}$$

  que denotaremos CA_3;

- métodos interpolatorios utilizando una ventana 5×5 puntos, es decir el producto tensorial del operador:

$$\begin{cases} (\mathcal{P}_{k-1}^k f^{k-1})_{2j-1} = -\frac{3}{128}f_{j-2}^{k-1} + \frac{22}{128}f_{j-1}^{k-1} + f_j^{k-1} - \frac{22}{128}f_{j+1}^{k-1} + \frac{3}{128}f_{j+2}^{k-1}, \\ (\mathcal{P}_{k-1}^k f^{k-1})_{2j} = \frac{3}{128}f_{j-2}^{k-1} - \frac{22}{128}f_{j-1}^{k-1} + f_j^{k-1} + \frac{22}{128}f_{j+1}^{k-1} - \frac{3}{128}f_{j+1}^{k-1}, \end{cases}$$

  que denotaremos como CA_5;

- los métodos de núcleo para medias en celda con función de pérdida $L_2(x,y)$.

En este capítulo vamos a mostrar sólo los resultados obtenidos con el núcleo `gaus` ya que es el más relevante. Obtendremos conclusiones analizando todos los resultados. Utilizaremos la distancia entre celdas dos dimensional $||\cdot||_p$ con $p = 1, 2, \infty$ para escoger las celdas que tomamos para aproximar. Así pues la notación será ${}^p\texttt{gaus}_m^n$ donde $n = 2, 3$ (si $n = 2$ la base de polinomios que utilizamos es $\{1, x, y, xy, x^2, y^2\}$, si $n = 3$ la base de polinomios es $\{1, x, y, xy, x^2, y^2, x^2y, xy^2, x^2y^2\}$) y m será el número de puntos.

Utilizaremos la siguiente cuantización:

$$\hat{d}_{i,j}^k = 2\varepsilon_k \left[\frac{d_{i,j}^k}{2\varepsilon_k}\right], \tag{3.39}$$

siendo d^k es el conjunto de detalles obtenidos en el paso k y donde

$$\varepsilon_{k-1} = \begin{cases} \frac{\varepsilon_k}{2}, & \textbf{si} \quad \varepsilon_k \geq 1/2; \\ 1/2, & \textbf{si} \quad \varepsilon_k < 1/2 \end{cases} \tag{3.40}$$

siendo ε_N dado y $[\cdot]$ es el entero obtenido por redondeo.

Para medir la capacidad de compresión del método contabilizamos el número de elementos no nulos (denotado por NNZ) siendo cuanto menor número mayor compresión.

Como podemos ver en las Tablas 3.3, 3.4 y 3.5 obtenemos mejores resultados utilizando el método de `gaus` ya que da un peso muy elevado a aquellas celdas muy próximas y da un peso muy pequeño (casi nulo) a aquellas que se encuentran lejos de la celda a aproximar. En aquellos métodos donde el núcleo da un peso más equitativo (núcleo `rect`) se obtienen peores resultados. Es necesario dar un valor muy alto al predictor en la celda $c_{i,j}^{k-1}$. Al utilizar el núcleo `gaus` este valor es muy alto en comparación con el resto de valores del operador predicción ya que la función decrece con rapidez. Además produce menor efecto *gibbs* en los contornos por:

1. La posibilidad de tomar más puntos sin tener que tomar otra base de polinomios con mayor grado.

2. La pérdida de tonalidad de la imagen en general al ser un método inconsistente (en comparación con los métodos de interpolación) que podemos observar en las Figuras 3.27, 3.29, 3.31 y 3.28, 3.30, 3.32. Esta pérdida de tonalidad se produce por la cuantización de todos los detalles (ver §2.3).

La elección de la métrica (1, 2 ó ∞) no afecta demasiado en los resultados obtenidos.

En definitiva los métodos basados en regresión local parece que producen mejores resultados que los interpolatorios gracias en gran medida a la utilización de la técnica (AY) y al peso dado por el núcleo elegido a las celdas cercanas a la que deseamos aproximar.

	$\varepsilon_N = 8$		$\varepsilon_N = 16$		$\varepsilon_N = 32$		$\varepsilon_N = 64$	
	NNZ	PSNR	NNZ	PSNR	NNZ	PSNR	NNZ	PSNR
CA_3	18881	34,00	8864	30,85	4138	27,77	1522	24,31
CA_5	18587	33,90	8632	30,72	4002	27,67	1482	24,22
${}^{2}\texttt{gaus}^{3}_{23}$	18644	34,14	8733	31,08	4085	28,09	1498	24,59
${}^{\infty}\texttt{gaus}^{3}_{25}$	18610	34,18	8735	31,14	4073	28,17	1500	24,63
${}^{1}\texttt{gaus}^{3}_{20}$	18570	34,10	8640	31,04	4032	28,12	1505	24,64
${}^{2}\texttt{gaus}^{2}_{23}$	18675	34,17	8767	31,15	4086	28,20	1497	24,67
${}^{\infty}\texttt{gaus}^{2}_{25}$	19553	34,09	9070	31,04	4178	28,12	1528	24,63
${}^{1}\texttt{gaus}^{2}_{20}$	18624	34,15	8702	31,09	4056	28,16	1505	24,67

Tabla 3.3: *Imagen* peppers*. Resultados utilizando multiresolución con método núcleo* gaus *con* $\varepsilon_N = 8, 16, 32, 64$*.*

3.10 Conclusiones y futuras líneas de investigación

En un esquema de multiresolución tenemos diferentes operadores que debemos construir con diferentes técnicas. El operador decimación (que ha de ser lineal y sobreyectivo) se basa en la elección del operador discretización. Habitualmente (ver, p.ej. [2, 3, 7, 15, 16, 53] y otros

	$\varepsilon_N = 8$		$\varepsilon_N = 16$		$\varepsilon_N = 32$		$\varepsilon_N = 64$	
	NNZ	PSNR	NNZ	PSNR	NNZ	PSNR	NNZ	PSNR
CA_3	6299	42,00	2838	36,70	1526	33,36	584	29,47
CA_5	5980	41,27	2766	36,59	1381	33,13	581	29,20
${}^{2}\texttt{gaus}^{3}_{23}$	6355	41,82	2846	37,03	1503	33,72	575	30,02
${}^{\infty}\texttt{gaus}^{3}_{25}$	6324	42,06	2821	37,02	1502	33,71	576	29,92
${}^{1}\texttt{gaus}^{3}_{20}$	6052	41,35	2806	36,81	1416	33,57	579	30,22
${}^{2}\texttt{gaus}^{2}_{23}$	6383	41,90	2831	37,12	1509	33,72	576	30,05
${}^{\infty}\texttt{gaus}^{2}_{25}$	6346	41,43	2921	36,80	1446	33,61	581	30,23
${}^{1}\texttt{gaus}^{2}_{20}$	6034	41,65	2814	36,92	1400	33,66	575	30,26

Tabla 3.4: *Imagen* geopa*. Resultados utilizando multiresolución con método núcleo* `gaus` *con* $\varepsilon_N = 8, 16, 32, 64$*.*

	$\varepsilon_N = 8$		$\varepsilon_N = 16$		$\varepsilon_N = 32$		$\varepsilon_N = 64$	
	NNZ	PSNR	NNZ	PSNR	NNZ	PSNR	NNZ	PSNR
CA_3	16324	34,30	7095	31,22	2719	28,28	883	25,77
CA_5	15758	34,38	6957	31,31	2676	28,31	853	25,72
${}^{2}\texttt{gaus}^{3}_{23}$	16148	34,53	7118	31,56	2725	28,67	880	26,12
${}^{\infty}\texttt{gaus}^{3}_{25}$	15950	34,47	6944	31,51	2677	28,65	855	26,09
${}^{1}\texttt{gaus}^{3}_{20}$	15678	34,41	6837	31,48	2647	28,64	841	26,08
${}^{2}\texttt{gaus}^{2}_{23}$	16039	34,44	6968	31,50	2680	28,66	854	26,11
${}^{\infty}\texttt{gaus}^{2}_{25}$	16846	34,33	7234	31,40	2733	28,61	879	26,12
${}^{1}\texttt{gaus}^{2}_{20}$	15822	34,43	6870	31,50	2660	28,66	852	26,12

Tabla 3.5: *Imagen* lena*. Resultados utilizando multiresolución con método núcleo* `gaus` *con* $\varepsilon_N = 8, 16, 32, 64$*.*

CA_3, PSNR: 27,77, NNZ: 4138

CA_5, PSNR: 27,67, NNZ: 4002

${}^{\infty}\mathtt{gaus}_{25}^{3}$, PSNR: 28,17, NNZ: 4073

${}^{\infty}\mathtt{gaus}_{25}^{2}$, PSNR: 28,12, NNZ: 4178

${}^{1}\mathtt{gaus}_{20}^{3}$, PSNR: 28,12, NNZ: 4032

${}^{1}\mathtt{gaus}_{20}^{2}$, PSNR: 28,16, NNZ: 4056

Figura 3.27: *Imágenes resultantes después de aplicar el algoritmo de compresión con la técnica (AY) y el operador predictor de los núcleos* ${}^{\infty}\mathtt{gaus}_{25}^{2,3}$ *y* ${}^{1}\mathtt{gaus}_{20}^{2,3}$ *para* peppers.

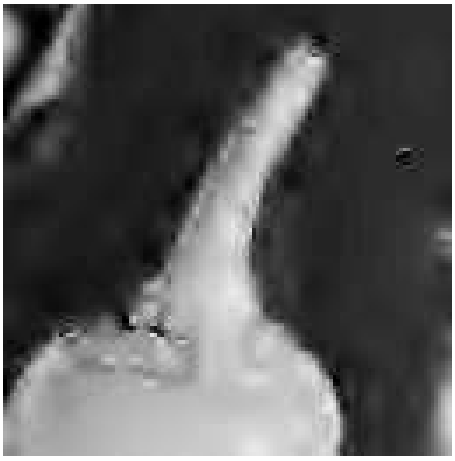
CA_3, PSNR: 27,77, NNZ: 4138

CA_5, PSNR: 27,67, NNZ: 4002

$^{\infty}\mathtt{gaus}^{3}_{25}$, PSNR: 28,17, NNZ: 4073

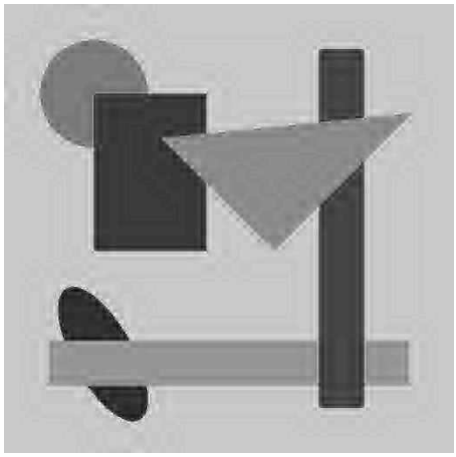
$^{\infty}\mathtt{gaus}^{2}_{25}$, PSNR: 28,12, NNZ: 4178

$^{1}\mathtt{gaus}^{3}_{20}$, PSNR: 28,12, NNZ: 4032

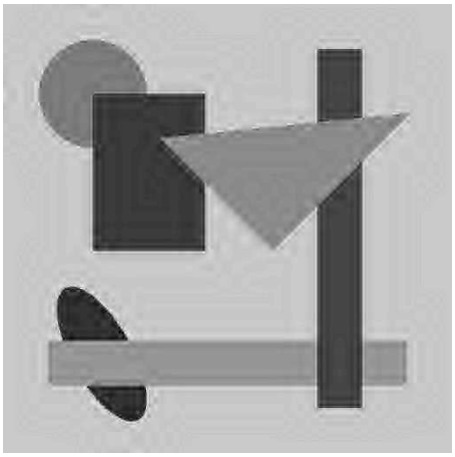


$^{1}\mathtt{gaus}^{2}_{20}$, PSNR: 28,16, NNZ: 4056

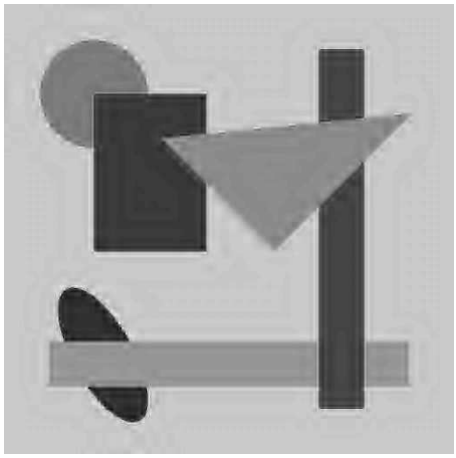
Figura 3.28: *Zoom de las imágenes resultantes después de aplicar el algoritmo de compresión con la técnica (AY) y el operador predictor de los núcleos* $^{\infty}\mathtt{gaus}^{2,3}_{25}$ *y* $^{1}\mathtt{gaus}^{2,3}_{20}$ *para* peppers.



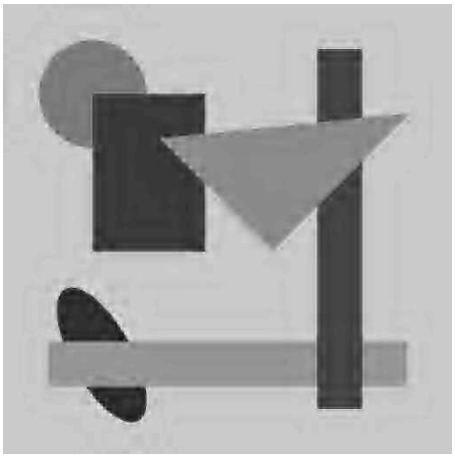
CA_3, PSNR: 33,36, NNZ: 1526



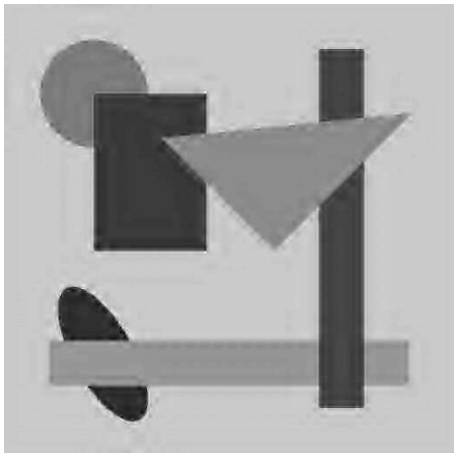
CA_5, PSNR: 33,13, NNZ: 1381



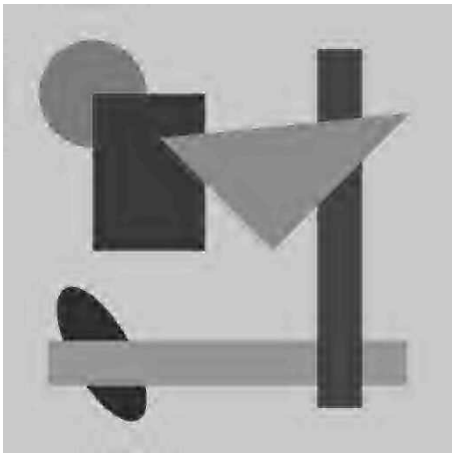
${}^{\infty}\mathtt{gaus}^{3}_{25}$, PSNR: 33,71, NNZ: 1502



${}^{\infty}\mathtt{gaus}^{2}_{25}$, PSNR: 33,61, NNZ: 1446

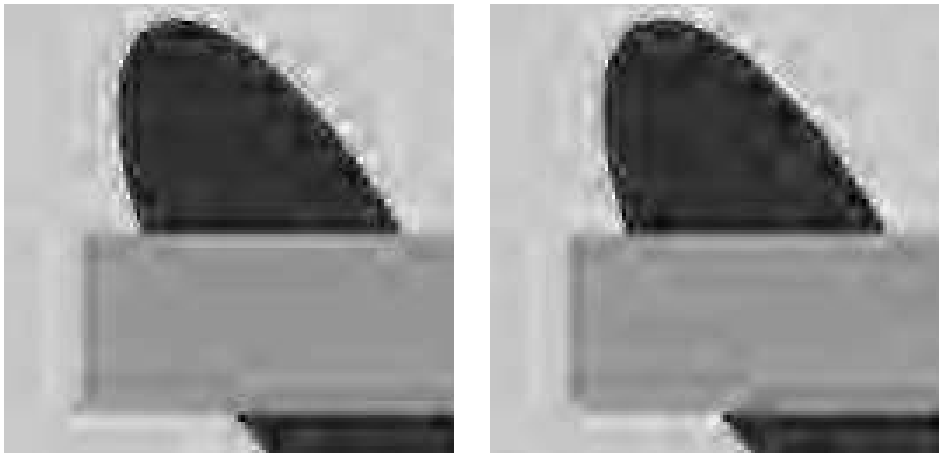


${}^{1}\mathtt{gaus}^{3}_{20}$, PSNR: 33,57, NNZ: 1416

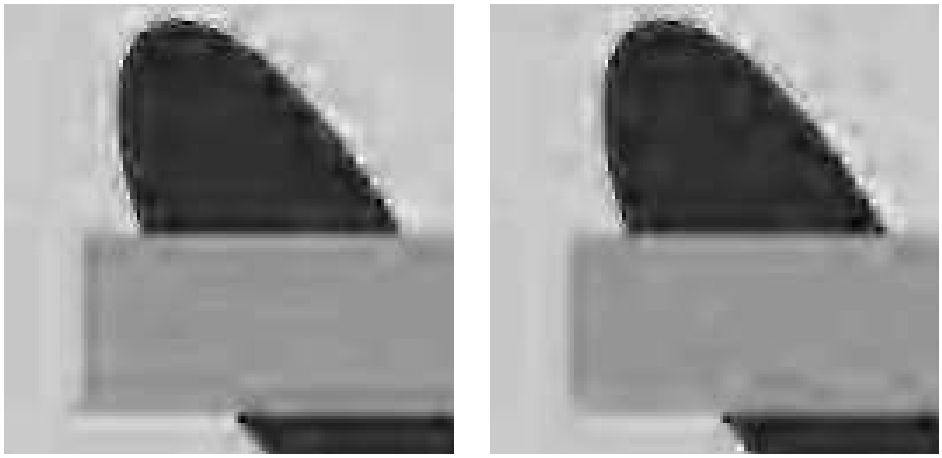


${}^{1}\mathtt{gaus}^{2}_{20}$, PSNR: 33,66, NNZ: 1400

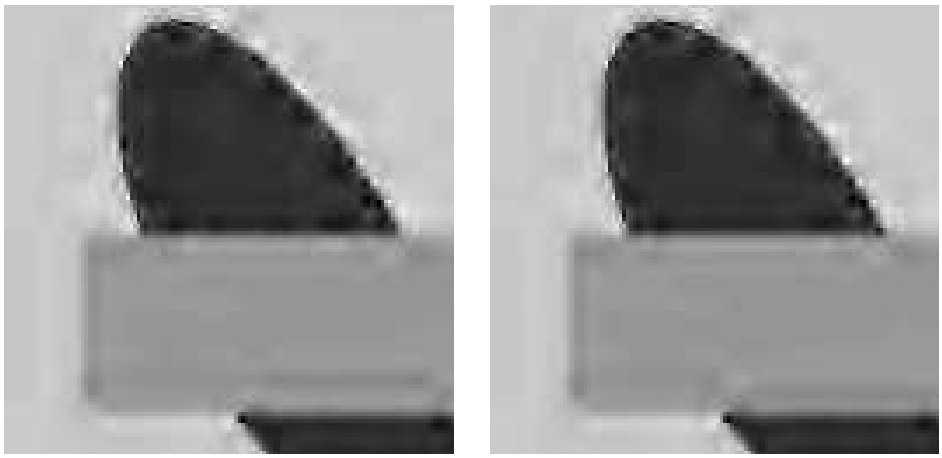
Figura 3.29: *Imágenes resultantes después de aplicar el algoritmo de compresión con la técnica (AY) y el operador predictor de los núcleos* ${}^{\infty}\mathtt{gaus}^{2,3}_{25}$ *y* ${}^{1}\mathtt{gaus}^{2,3}_{20}$ *para* geopa.



CA_3, PSNR: 33,36, NNZ: 1526 CA_5, PSNR: 33,13, NNZ: 1381



$^{\infty}\texttt{gaus}^{3}_{25}$, PSNR: 33,71, NNZ: 1502 $^{\infty}\texttt{gaus}^{2}_{25}$, PSNR: 33,61, NNZ: 1446



$^{1}\texttt{gaus}^{3}_{20}$, PSNR: 33,57, NNZ: 1416 $^{1}\texttt{gaus}^{2}_{20}$, PSNR: 33,66, NNZ: 1400

Figura 3.30: *Zoom de las imágenes resultantes después de aplicar el algoritmo de compresión con la técnica (AY) y el operador predictor de los núcleos* $^{\infty}\texttt{gaus}^{2,3}_{25}$ *y* $^{1}\texttt{gaus}^{2,3}_{20}$ *para* geopa.

CA_3, PSNR: 28,28, NNZ: 2719

CA_5, PSNR: 28,31, NNZ: 2676

${}^{\infty}\mathtt{gaus}^{2}_{25}$, PSNR: 28,65, NNZ: 2676

${}^{\infty}\mathtt{gaus}^{3}_{25}$, PSNR: 28,61, NNZ: 2733

${}^{1}\mathtt{gaus}^{2}_{20}$, PSNR: 28,64, NNZ: 2646

${}^{1}\mathtt{gaus}^{3}_{20}$, PSNR: 28,66, NNZ: 2660

Figura 3.31: *Imágenes resultantes después de aplicar el algoritmo de compresión con la técnica (AY) y el operador predictor de los núcleos* ${}^{\infty}\mathtt{gaus}^{2,3}_{25}$ *y* ${}^{1}\mathtt{gaus}^{2,3}_{20}$ *para* lena.

CA_3, PSNR: 28,28, NNZ: 2719

CA_5, PSNR: 28,31, NNZ: 2676

${}^{\infty}\mathtt{gaus}^{2}_{25}$, PSNR: 28,65, NNZ: 2676

${}^{\infty}\mathtt{gaus}^{3}_{25}$, PSNR: 28,61, NNZ: 2733

${}^{1}\mathtt{gaus}^{2}_{20}$, PSNR: 28,64, NNZ: 2646

${}^{1}\mathtt{gaus}^{3}_{20}$, PSNR: 28,66, NNZ: 2660

Figura 3.32: *Zoom de las imágenes resultantes después de aplicar el algoritmo de compresión con la técnica (AY) y el operador predictor de los núcleos* ${}^{\infty}\mathtt{gaus}^{2,3}_{25}$ *y* ${}^{1}\mathtt{gaus}^{2,3}_{20}$ *para* lena.

muchos) la elección del operador discretización se basa en la convolución de una función f con alguna de las potencias de la función $\omega_0(x) = \chi_{[-\frac{1}{2},\frac{1}{2}]}(x)$ como vimos en la §2.2.2. En este capítulo fijamos el operador discretización (tomamos las potencias $n = 0, 1, 2$ de la función $\chi_{[0,1]}(x)$) y presentamos una nueva técnica para construir un operador predicción basada en regresión local polinómica. Introducimos por tanto un problema de la forma:

$$\arg\min_{g\in\mathcal{K}} \sum_{i=1}^{n} K_\lambda(f_i^{k-1}, g(x_i^{k-1}))L(f_i^{k-1}, g(x_i^{k-1})),$$

siendo $\{x_i^{k-1}\}_{i\in\mathcal{M}}$, $\mathcal{M} \subset \mathbb{N}$, un posible mallado y f^{k-1} son los valores de la función f utilizando el operador discretización ya definido en el nivel de resolución $k-1$; K_λ es una función núcleo dependiente de un ancho de banda λ que determinará el número de puntos que utilizamos para la regresión; la función $L(x, y)$ que es una función de pérdida que marca la distancia entre el valor real y la aproximación, en este capítulo hemos tratado $L_p(x, y) = |x - y|^p$ y la norma del espacio ℓ^∞; por último la elección de la clase $\mathcal{K}$ donde minimizamos el funcional, en este capítulo solo hemos tratado polinomios, i.e. $\mathcal{K} = \Pi_n^r(\mathbb{R})$, con $n = 1, 2$.

Por tanto hemos construido gran cantidad de métodos de multiresolución basándonos en los siguientes parámetros:

Función peso K_λ Hemos propuesto diferentes funciones peso basándonos en la literatura habitual en regresión local (ver [56, 70]), cada una ofrece diferentes operadores predicción. Una posible línea de investigación sería probar con otras funciones peso diferentes a las presentadas en esta tesis.

El ancho de banda λ Al utilizar interpolación polinómica en el diseño del operador predicción (capítulo 2 §2.2.3) necesitamos el mismo número de puntos que el grado del polinomio. En este caso no es necesario, el parámetro λ determina la cantidad de puntos que utilizamos para aproximar con independencia del grado del polinomio utilizado. Esto puede ser de gran utilidad para aprovechar cierta información útil de algunos puntos sin subir el grado del polinomio aproximante.

Función de pérdida $L(x, y)$ Hemos generalizado la norma ℓ^2 al conjunto de todas las normas ℓ^p con $p = 1, 2, \ldots$ utilizando para $p = 1$ el método propuesto por T. Chan y P. Mulet en [26] y para el resto

de normas programación convexa (ver [20, 49, 50]). Podríamos generalizarla también en dos dimensiones. *A priori* parece que obtendremos resultados interesantes.

Generalización en dos dimensiones Para extender nuestros resultados a dos dimensiones presentamos diferentes normas para definir la distancia entre dos celdas en las funciones peso. La elección de estas normas produce diferentes operadores predicción.

Clase de funciones, $\mathcal{K}$ Hemos utilizado tan sólo polinomios. Otra posible línea de investigación es el cambio en la clase de funciones donde minimizamos el funcional. Esto podría ofrecer ventajas si tenemos funciones irregulares.

Así pues, presentamos distintas variables que combinándolas nos dan diferentes métodos de predicción cuyos resultados numéricos hemos visto en la sección anterior.

Uno de los problemas que plantea el método es la poca adaptabilidad del parámetro λ, este problema podría ser una línea de investigación, escoger un parámetro dependiente de los valores f^k. Por otra parte, en la frontera encontramos también un problema de aproximación, debido a que se toma menor cantidad de puntos (ver más detalle en [56]). Esto lo resolvemos realizando interpolación descentrada, otra solución es suponer periodicidad en la función f.

Otra posible línea de investigación sería combinar el método de núcleo con métodos no lineales como ENO y WENO.

4

Operadores basados en técnicas estadísticas de aprendizaje: Multiresolución de aprendizaje

Hoy en día la teoría estadística de aprendizaje juega un papel importante en muchas áreas de ciencia, industria y finanzas. Algunos ejemplos de problemas de aprendizaje son:

Biomatemática

- Predecir si un paciente que fue ingresado debido a un ataque de corazón tendrá un segundo ataque. La predicción estará basada en la dieta y los análisis clínicos del paciente.

- Estimar la media de glucosa en sangre de una persona diabética mediante el espectro de absorción en el infrarrojo de su sangre.
- Identificar los factores de riesgo para contraer cáncer basado en variables clínicas y demográficas.

Economía

- Predecir el precio de un *stock* a los 6 meses, en base a los datos de economía y las bases de la compañía.

Análisis de formas

- Identificar los números en una carta escrita desde una imagen digital.

En la multiresolución explicada en el capítulo 2 veíamos que al decimar un conjunto de datos perdíamos parte de la información original que recuperábamos en el espacio de detalles (que era el núcleo del operador decimación). Nosotros proponemos un cambio en la forma de hallar el operador predicción basándonos en la utilización de técnicas estadísticas de aprendizaje con el fin de encontrar el mejor filtro para un conjunto de entrenamiento determinado. Al estudiar esta nueva orientación del problema se descubren distintas posibilidades para su utilización en compresión de señales e imágenes digitales.

Estos estudios se transforman en un primer trabajo en [94]. En este capítulo introduciremos este tipo de multiresolución, veremos algunas mejoras e intentaremos ver las ventajas y desventajas que tenemos al utilizar este método.

4.1 Introducción

La ciencia del aprendizaje es de gran importancia en Estadística, clasificación de datos e Inteligencia Artificial junto con otras áreas de la Ingeniería y otras disciplinas.

Los elementos que conforman un problema de aprendizaje son tres:

1. Un conjunto de vectores de **entrada** $X^1, \ldots, X^l$. Estos valores pueden ser de distinto tipo (vectores reales o naturales) y pueden proceder de distintos interpretaciones de los datos iniciales de un problema real.

Es decir, podemos considerarlos como ciertos datos de economía o los datos de cierto estudio químico (ver p. ej. [70, 56, 57]). En nuestro caso estos datos vendrán de la discretización de una función f (ver el capítulo 2).

II. Un conjunto de valores **respuesta** y^i para cada valor de entrada X^i. Entendemos que estos valores responden a cierta función $z(X)$. Así tendríamos,

$$y^i = z(X^i) + \epsilon_i,$$

con ϵ_i independientes e idénticamente distribuidos con media cero.

III. Una función $\hat{z}$ que sea capaz de reproducir cualquier otro valor de la misma naturaleza que los datos de entrada que llamaremos **predictor de aprendizaje**.

Durante el proceso de aprendizaje, el predictor de aprendizaje (PA) trabaja con los pares (X^i, y^i), $i = 1, \ldots, l$ (el conjunto de entrenamiento). Después de entrenar, para cualquier vector de entrada X el predictor debe devolver un valor $\hat{y}$. El objetivo es devolver un valor $\hat{y}$ muy próximo a la respuesta de la función $z(X)$.

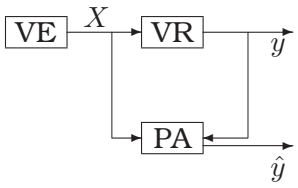


Figura 4.1: *Modelo de aprendizaje desde una muestra, donde* VE *son valores de entrada,* VR *son los valores respuesta y* PA *es el predictor de aprendizaje.*

Supongamos, como en el capítulo 2, que tenemos un esquema de multiresolución formado por una sucesión finita de espacios $\{V^k\}_k$ y de operadores decimación fijos $\{\mathcal{D}_k^{k-1}\}_k$. Supongamos, además, que $V^{k-1} = \mathcal{D}_k^{k-1} V^k$. Habitualmente (como se ve en el capítulo 2 y p.ej. en [2, 14, 13, 15, 16, 53]) los datos del nivel k se hallan con un operador predicción basado en interpolación polinómica o, como hemos visto en el capítulo 3, con regresión local. En cualquier caso no utilizamos en la construcción de este operador la información que poseemos del nivel k sino que toda esta información está contenida en los valores de los detalles

(recordamos $e^k = f^k - \mathcal{P}_{k-1}^k f^{k-1}$) que formaban el núcleo del operador decimación, que denotábamos como $\mathcal{N}(\mathcal{D}_k^{k-1})$. Por tanto teníamos (ver [53] o capítulo VII, §7.6 de [72]):

$$V^k = V^{k-1} \oplus \mathcal{N}(\mathcal{D}_k^{k-1}).$$

La compresión de un conjunto de valores (véase imágenes o señales digitales) se basa en que el número de detalles almacenados distintos de cero sea pequeño (cuanto menor sea este valor mayor es la compresión), teniendo en cuenta que aquellos detalles pequeños que no afectan a la reconstrucción de la señal se truncan a cero. El problema (ver [13]) por tanto se reduce a

> Conociendo f^{k-1} encontrar una buena aproximación a f^k.

Esto nos lleva a preguntarnos: ¿por qué no utilizar los datos del nivel k para construir nuestro operador predicción? Es decir, ¿por qué si sabemos el resultado que debemos obtener no "buscamos" el(los) mejor(es) filtro(s) para pasar del nivel $k-1$ a k minimizando el conjunto de detalles? Planteamos por tanto el siguiente problema:

$$\underset{\mathcal{P}_{k-1}^k \in \mathcal{K}}{\arg\min}\, \mathcal{T}(f^k, \mathcal{P}_{k-1}^k(f^{k-1})), \tag{4.1}$$

donde $\mathcal{K}$ es una clase de funciones, no necesariamente lineales y $\mathcal{T}$ es una función que marca la distancia entre los valores del espacio V^k y la predicción. Transformamos el problema en:

> Conociendo f^{k-1} y f^k, encontrar una buena aproximación a f^k utilizando tan solo los valores de f^{k-1} con $\Upsilon(\mathcal{N}(\mathcal{D}_k^{k-1}))$,

donde $\Upsilon(\mathcal{N}(\mathcal{D}_k^{k-1}))$ son ciertas condiciones en el núcleo del operador discretización, p. ej. que la mayoría de sus componentes sean cero (o cercanas a cero).

El estudio del problema (4.1) con sus distintas variables, su desarrollo, sus ventajas y desventajas conforma el contenido de este capítulo.

4.2 Multiresolución de aprendizaje 1D

En primer lugar veamos cuales son las componentes necesarias en un problema de aprendizaje relacionándolas con los datos que tenemos en un esquema de multiresolución.

Supongamos que tenemos una serie de espacios $\{V^k\}_{k=1}^N$ con un conjunto de operadores decimación ya elegido $\{\mathcal{D}_k^{k-1}\}_{k=1}^N$ (determinado por un operador discretización, ver capítulo 2), así denotaremos como $f_i^k = (\mathcal{D}_k f)_i$ con $i \in \mathcal{M}_k \subset \mathbb{N}$ y el vector $f^k = (f_i^k)_{i\in\mathcal{M}_k}$. Además, si $r, s \in \mathbb{N}$ definimos los vectores:

$$\mathcal{S}^{r,s}(f_i^k) = (f_{i-r}^k, \ldots, f_{i+s}^k),\ i \in \mathcal{M}_k$$

que son los valores de la discretización en los valores próximos a f_i^k.

Nota 4.1. *El problema que presenta la frontera será tratado utilizando un esquema de interpolación polinómica segmentaria descentrado dependiendo del número de celdas que utilicemos.*

Nota 4.2. *Explicaremos el problema bajo el contexto de valores puntuales. Sería análago para cualquier tipo de discretización.*

Así pues traducimos los elementos del problema de aprendizaje a multiresolución:

I. *Un conjunto de vectores de entrada* $X^1, \ldots, X^l$. En nuestro caso tomamos los vectores $\mathcal{S}^{r,s}(f_i^{k-1})$ con $i \in \mathcal{M}_k$, i.e $X^i = \mathcal{S}^{r,s}(f_i^{k-1})$. Tomamos los elementos que deseamos utilizar del espacio V^{k-1} para determinar V^k.

II. *Un conjunto de valores respuesta* y^i *para cada valor de entrada* X^i. En multiresolución será cada uno de los valores del nivel k, es decir $y^i = f_{2i-1}^k$.

III. *Una función* $\hat{z}$ *que sea capaz de reproducir cualquier otro valor de la misma naturaleza que los datos de entrada que llamaremos predictor de aprendizaje.* Así la función a hallar es un operador predicción, $\hat{\mathcal{P}}_{k-1}^k$, capaz de, utilizando los datos de f^{k-1}, aproximar a los valores de f^k con la mejor precisión posible.

El problema de aprendizaje, si $L(x,y)$ es una función de pérdida, se resuelve (ver [90]) minimizando el funcional definido en una clase de funciones determinada $\mathcal{K}$:

$$R_{e,L}(g) = \frac{1}{|\mathcal{M}_k|} \sum_{i \in \mathcal{M}_k} L(f^k_{2i-1}, g(\mathcal{S}^{r,s}(f^{k-1}_i))),$$

por tanto, nuestro operador predicción sería,

$$\hat{\mathcal{P}}^k_{k-1} = \arg\min_{g \in \mathcal{K}} R_{e,L}(g) = \arg\min_{g \in \mathcal{K}} \sum_{i \in \mathcal{M}_k} L(f^k_{2i-1}, g(\mathcal{S}^{r,s}(f^{k-1}_i))), \tag{4.2}$$

donde $\hat{\mathcal{P}}^k_{k-1} : \mathcal{S}^{r,s}(V^{k-1}) \to \mathbb{R}$ con

$$\mathcal{S}^{r,s}(V^{k-1}) = V^{k-1} \times \underset{r+s+1}{\ldots} \times V^{k-1}.$$

Así definiremos el operador predictor $\mathcal{P}^k_{k-1} : V^{k-1} \to V^k$ como

$$\mathcal{P}^k_{k-1} f^{k-1} = (\hat{\mathcal{P}}^k_{k-1} \mathcal{S}^{r,s}(f^{k-1}_1), \ldots, \hat{\mathcal{P}}^k_{k-1} \mathcal{S}^{r,s}(f^{k-1}_{|\mathcal{M}_k|})).$$

Para no recargar la notación siempre utilizaremos $\hat{\mathcal{P}}^k_{k-1}$ ya que los filtros son únicos pues son iguales para cada píxel $i \in \mathcal{M}_k$.

Ejemplo 4.1. *Veamos cómo definiríamos el operador en el caso de valores puntuales en una dimensión. Así supongamos como en la §2.2.2 que tenemos el operador decimación definido como:*

$$(\mathcal{D}^{k-1}_k f^k)_j = f^k_{2j}, \quad j = 0, \ldots, J_{k-1},$$

entonces tendremos que minimizar el siguiente funcional:

$$\hat{\mathcal{P}}^k_{k-1} = \arg\min_{g \in \mathcal{K}} \sum_{i=r}^{J_{k-1}-s} L(f^k_{2i-1}, g(\mathcal{S}^{r,s}(f^{k-1}_i))),$$

entonces definimos el operador predicción como

$$\begin{aligned} &(\mathcal{P}^k_{k-1} f^{k-1})_{2j} = f^{k-1}_j, \quad 0 \le j \le J_{k-1}; \\ &(\mathcal{P}^k_{k-1} f^{k-1})_{2j-1} = \hat{\mathcal{P}}^k_{k-1}(\mathcal{S}^{r,s}(f^{k-1}_j)), \quad r \le j \le J_{k-1} - s. \end{aligned}$$

En la frontera definiremos el operador predicción utilizando el esquema descentrado basado en interpolación segmentaria §2.2.3.

Tenemos varias componentes en nuestro problema:

La función de pérdida $L(x, y)$ Esta función de pérdida, al igual que en el capítulo anterior (capítulo 3) marca la distancia entre el valor esperado, f^k_{2i-1} y el vector, $\mathcal{S}^{r,s}(f^{k-1})$ con el que vamos a aproximar dicho valor.

La clase de funciones $\mathcal{K}$ Podemos tomar cualquier clase de funciones (siempre y cuando podamos minimizar el funcional) lo que nos permite una mayor elección de funciones (por ejemplo funciones no lineales). En la práctica no hemos encontrado un conjunto que aproxime mejor que las funciones lineales.

Algunas modificaciones con la multiresolución utilizada hasta ahora son:

1. Para la definición del operador predicción no es necesario conocer el operador discretización. Habitualmente en la literatura se utiliza una aproximación a la función f por medio de un polinomio $p(x)$ y después se discretiza. En nuestro caso no es necesario conocer la naturaleza de los datos para definir el operador predicción.

2. Al igual que sucedía con los métodos núcleo podemos escoger el número de elementos en f^{k-1} que deseemos (teniendo en cuenta que a mayor número mayor será la dificultad para resolver el problema), i. e., tenemos libertad en la elección de los valores r y s.

3. Para hallar el predictor tenemos información de los valores en V^k, por tanto debemos **almacenar** este operador en cada paso de la multiresolución.

Por este último punto tendríamos (utilizando la misma notación que en el capítulo 2) $f^k \equiv (f^{k-1}, \hat{\mathcal{P}}^k_{k-1}, d^k)$. Se podría pensar que estamos almacenando mayor número de elementos, sin embargo el operador predicción está vinculado a la minimización de los detalles (y su posible anulación). **La ventaja radica en que el número de elementos que guardemos de la predicción sea inferior al número de elementos que anulemos eligiendo este operador.** Si repetimos este proceso en cada nivel de resolución tenemos la equivalencia:

$$f^N \equiv (f^0, \hat{\mathcal{P}}^1_0, d^1, \hat{\mathcal{P}}^2_1, d^2, \ldots, \hat{\mathcal{P}}^N_{N-1}, d^N).$$

Veamos cómo cambia el esquema y el algoritmo de descomposición con la modificación producida por el aprendizaje.

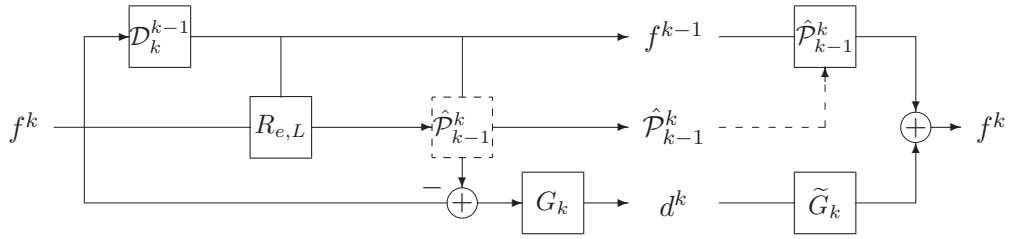


Figura 4.2: *Descomposición sin información redundante utilizando la codificación del error de predicción utilizando multiresolución de aprendizaje.*

Algoritmo 4.1. *Transformación directa utilizando LM* $f^N \to Mf^N = (f^0, \hat{\mathcal{P}}_0^1, d^1, \ldots, \hat{\mathcal{P}}_{N-1}^N, d^N)$

for $k = N, \ldots, 1$
$f^{k-1} = \mathcal{D}_k^{k-1} f^k$
$\hat{\mathcal{P}}_{k-1}^k = \arg\min_{g \in \mathcal{K}} R_{e,L}(g)$
$d^k = G_k(f^k - \hat{\mathcal{P}}_{k-1}^k f^{k-1})$
end

Para comprender mejor el método veamos un ejemplo.

Hacemos este ejemplo bajo el contexto de valores puntuales. Tomamos una función $f \in \mathcal{F} = \mathcal{B}[0, 32]$, el espacio de funciones acotadas en el intervalo $[0, 32]$.

Tomaremos $N = 5$, es decir, $J_5 = 32$ puntos y tenemos los siguientes datos iniciales:

$$f_t^5 = \begin{cases} 64, & t = 2m \text{ con } m = 0, \ldots, 16; \\ 16, & t = 2m + 1 \text{ con } m = 0, \ldots, 15. \end{cases} \tag{4.3}$$

Si decimamos los datos obtenemos $f_j^4 = 64 \ \forall \ j = 1, \ldots 16$. Si interpolamos los datos (ver ejemplos §2.2.3) obtenemos la constante 64 y por tanto $e_j^k = -48 \ \forall \ j = 1, \ldots 16$.

Veamos el caso particular de un esquema LM cuyo espacio donde minimizar será el conjunto de funciones lineales de cuatro variables, i.e. $\mathcal{K} = \Pi_4^1(\mathbb{R})^*$[1] (con esta elección de las funciones, si queremos tomar un *stencil* centrado debemos escoger $s = 1$ y $r = 2$), que aproxime a los

[1] Indicaremos $\Pi_m^n(\mathbb{R})^*$ al espacio de polinomios de m variables de grado n sin término

valores deseados utilizando la norma vectorial ℓ^2, así, tomamos $L_2(x,y) = (x-y)^2$. Por tanto, nuestro problema es

$$\hat{\mathcal{P}}^k_{k-1} = \underset{g\in\Pi^1_4(\mathbb{R})}{\arg\min} R_{e,L}(g) = \underset{g\in\Pi^1_4(\mathbb{R})}{\arg\min} \sum_{i=2}^{31} (f^5_{2i-1} - g(\mathcal{S}^{2,1}(f^4_i)))^2,$$

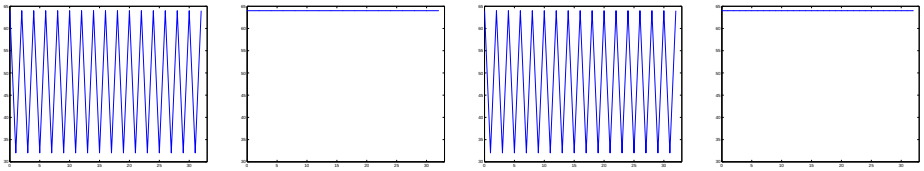
así tendríamos

$$\hat{\mathcal{P}}^k_{k-1}\mathcal{S}^{2,1}(f^4_j) = \hat{\beta}_{-2}f^4_{j-2} + \hat{\beta}_{-1}f^4_{j-1} + \hat{\beta}_0 f^4_j + \hat{\beta}_1 f^4_{j+1},$$

con

$$\begin{aligned}\hat{\beta} &= \underset{\beta\in\mathbb{R}^4}{\min} \sum_{i=2}^{31} (f^5_{2i-1} - \mathcal{S}^{2,1}(f^4_i)\beta^T)^2 \\ &= \underset{\beta\in\mathbb{R}^4}{\min} \sum_{i=2}^{31} (f^5_{2i-1} - \beta_{-2}f^4_{i-2} + \beta_{-1}f^4_{i-1} + \beta_0 f^4_i + \beta_1 f^4_{i+1})^2.\end{aligned}$$

Es fácil ver que es un problema de mínimos cuadrados sin solución única, un posible resultado sería $\hat{\beta} = \left(0, \frac{1}{4}, 0, 0\right)$. Y por tanto nuestro operador predictor es $(\hat{\mathcal{P}}^5_4 f^4)_{2i-1} = \frac{f^4_{i-1}}{4}$ obteniendo la función tal cual.



f^5 f^4 $\hat{\mathcal{P}}^5_4 f^4$ $\mathcal{P}^5_4 f^4$

Figura 4.3: *Comparación en un ejemplo sencillo entre un operador predictor usando LM, $\hat{\mathcal{P}}^k_{k-1}$, y un operador, $\mathcal{P}^k_{k-1}$, utilizando interpolación de los datos.*

4.2.1 La clase de funciones. Orden y control del error de un esquema de multiresolución de aprendizaje (LM)

Comenzamos esta sección hablando del orden (Definición 2.2) probando el siguiente teorema:

independiente. Ejemplo:

$$\Pi^1_4(\mathbb{R})^* = \{f : \mathbb{R}^4 \to \mathbb{R} / f(X) = X\beta^T, \beta \in \mathbb{R}^4\}$$

Teorema 4.1. *En el contexto de valores puntuales (análogo para medias en celda y medias* hat*) si* $\Pi^1_{r+s+1}(\mathbb{R})^* \subseteq \mathcal{K}$*, y si tomamos* $(\mathcal{S}^{r+s}(f_i^{k-1}), f_{2i-1}^k)_{i\in\mathcal{M}}$ *como conjunto de entrenamiento entonces el orden del esquema de multiresolución (según la Definición 2.2) es al menos* $r+s+1$ *con independencia de la función pérdida,* $L(x,y)$*, que utilicemos.*

Demostración Sea $p(x) \in \Pi_1^{r+s}(\mathbb{R})$ y $p^k = \mathcal{D}_k p$ (siendo $\mathcal{D}$ el operador discretización en el contexto de valores puntuales), si tomamos los puntos $\mathcal{S}_i^{r+s} = \{x_{i-r}, \ldots, x_{i+s}\}$ para construir el predictor basándonos en interpolación polinómica segmentaria obtenemos el operador lineal $\mathcal{P}_{k-1}^k$:

$$(\mathcal{P}_{k-1}^k f^{k-1})_{2i-1} = \sum_{t=-r}^{s} a_t f_{i+t}^{k-1}, \text{ con } a_t \in \mathbb{R}, t = -r, \ldots, s$$

que es de orden $r+s+1$ como vimos en §2.2.3, así

$$\sum_{i\in\mathcal{M}} L(p_{2i-1}^k, (\mathcal{P}_{k-1}^k p^{k-1})_{2i-1}) = 0,$$

definimos $(\hat{\mathcal{P}}_{k-1}^k f^{k-1})_{2i-1} : \mathcal{S}^{r,s}(f_i^{k-1}) \to \mathbb{R}$ como

$$(\hat{\mathcal{P}}_{k-1}^k f^{k-1})_{2i-1} = \sum_{t=-r}^{s} a_t f_{i+t}^{k-1} = (\mathcal{P}_{k-1}^k f^{k-1})_{2i-1};$$

$\hat{\mathcal{P}}_{k-1}^k$ pertenece a $\Pi^1_{r+s+1}(\mathbb{R})^* \subseteq \mathcal{K}$. ■

Para poder guardar los operadores predicción tenemos que cambiar su representación digital como veremos en la §4.3.2. Veamos que ha de cumplir este cambio en el operador predicción para tener control del error cuando aplicamos el algoritmo de recontrucción.

Si controlamos la cuantización en cada uno de los niveles de la multiresolución controlamos el error final que obtendremos al reconstruir la señal o imagen. En el siguiente resultado $\hat{f}$ denota, como es habitual, una posible versión perturbada de f.

Teorema 4.2. *Si $\mathcal{K}$ es el espacio de funciones lineales entonces para cualquier par de elementos $f^N, \hat{f}^N \in V^N$ tal que existe C_{V_N} que cumple*

$$||g^N|| \leq C_{V^N}, \forall\, g^N \in V^N,$$

cuyas descomposiciones utilizando el esquema visto anteriormente son $\{f^0, \mathcal{P}_0^1, d^1, \ldots, \mathcal{P}_{N-1}^N, d^N\}$ y $\{\hat{f}^0, \hat{\mathcal{P}}_0^1, \hat{d}^1, \ldots, \hat{\mathcal{P}}_{N-1}^N, \hat{d}^N\}$ cumpliendo que

$$\begin{aligned} &||d^k - \hat{d}^k|| \leq \frac{\varepsilon_k}{2}, \quad k = 1, \ldots, N; \\ &||\mathcal{P}_{k-1}^k - \hat{\mathcal{P}}_{k-1}^k|| \leq \frac{\varepsilon_k}{2C_{V^N}||\mathcal{D}_k^{k-1}||}, \quad k = 1, \ldots, N; \\ &||f^0 - \hat{f}^0|| \leq \varepsilon_0; \end{aligned} \tag{4.4}$$

entonces si $K = \max_{k=1,\ldots,N}\{||\mathcal{P}_{k-1}^k||\}$ y

$$\varepsilon_k = \varepsilon K^{k-N} q^{N-k}, \quad 0 < q < 1,$$

entonces

$$||f^N - \hat{f}^N|| \leq \frac{\varepsilon}{1-q} \tag{4.5}$$

con independencia de la función peso $L(x, y)$ elegida.

Demostración Por la linealidad de los operadores decimación y predicción existen tres matrices, $\mathbf{B}_k^{k-1} \in \mathbb{R}^{J_{k-1} \times J_k}$, $\mathbf{A}_{k-1}^k \in \mathbb{R}^{J_k \times J_{k-1}}$ y $\hat{\mathbf{A}}_{k-1}^k \in \mathbb{R}^{J_k \times J_{k-1}}$ tales que:

$$\begin{aligned} &\mathcal{D}_k^{k-1} f^k = \mathbf{B}_k^{k-1} f^k, \quad \forall\, f^k \in V^k, \\ &\mathcal{P}_{k-1}^k f^{k-1} = \mathbf{A}_{k-1}^k f^{k-1}, \quad \forall\, f^{k-1} \in V^{k-1}, \\ &\hat{\mathcal{P}}_{k-1}^k \hat{f}^{k-1} = \hat{\mathbf{A}}_{k-1}^k \hat{f}^{k-1}, \quad \forall\, \hat{f}^{k-1} \in V^{k-1}. \end{aligned}$$

Sea $0 < j \leq N$, sea $K = \max_{j=1,\ldots,N} ||\mathbf{A}_{j-1}^{j}||$ entonces por hipótesis del teorema tenemos que

$$\begin{aligned}
||f^j - \hat{f}^j|| &= ||(\mathcal{P}_{j-1}^{j} f^{j-1} + d^j) - (\hat{\mathcal{P}}_{j-1}^{j} \hat{f}^{j-1} + \hat{d}^j)|| \\
&= ||\mathcal{P}_{j-1}^{j} f^{j-1} + \mathcal{P}_{j-1}^{j} \hat{f}^{j-1} - \mathcal{P}_{j-1}^{j} \hat{f}^{j-1} - \hat{\mathcal{P}}_{j-1}^{j} \hat{f}^{j-1} + d^j - \hat{d}^j|| \\
&\leq ||\mathbf{A}_{j-1}^{j}(f^{j-1} - \hat{f}^{j-1})|| + ||\mathbf{A}_{j-1}^{j} - \hat{\mathbf{A}}_{j-1}^{j}||||\hat{f}^{j-1}|| + ||d^j - \hat{d}^j|| \\
&\leq ||\mathbf{A}_{j-1}^{j}(f^{j-1} - \hat{f}^{j-1})|| + ||\mathbf{A}_{j-1}^{j} - \hat{\mathbf{A}}_{j-1}^{j}||||\mathcal{D}_{j}^{j-1} \hat{f}^{j}|| + ||d^j - \hat{d}^j|| \\
&\leq ||\mathbf{A}_{j-1}^{j}(f^{j-1} - \hat{f}^{j-1})|| + \frac{\varepsilon_j}{2} + \frac{\varepsilon_j}{2} \\
&\leq ||\mathbf{A}_{j-1}^{j}||||f^{j-1} - \hat{f}^{j-1}|| + \varepsilon_j.
\end{aligned}$$

Así eligiendo $\varepsilon_k = \varepsilon K^{k-N} q^{N-k}$ se cumple que

$$||f^N - \hat{f}^N|| = K^N ||f^0 - \hat{f}^0|| + \sum_{k=1}^{N} K^{N-k} \varepsilon_k = \varepsilon \sum_{k=0}^{N} q^{N-k} = \varepsilon \frac{1 - q^N}{1 - q} < \frac{\varepsilon}{1 - q}.$$

■

Ejemplo 4.2. *Sea el conjunto de funciones lineales de $r + s + 1$ variables $\mathcal{K} = \Pi_{r+s+1}^{1}(\mathbb{R})^* = \{f / f(X) = X\beta^T, \beta \in \mathbb{R}^{r+s+1}\}$, entonces por los teoremas 4.1 y 4.2 tendrá orden $r + s$ y control del error para las normas $|| \cdot ||_p$ con $p = 1, 2, \infty$.*

En esta tesis **solo** utilizaremos como clase de funciones $\mathcal{K} = \Pi_{r+s+1}^{1}(\mathbb{R})^*$, es decir, polinomios $r + s + 1$-dimensionales de grado 1. Con esta clase tenemos probado el orden y el control del error. Hemos hecho experimentos con otras clases de funciones no lineales pero no hemos obtenidos buenos resultados.

4.2.2 Función de pérdida

La elección de la función $L(x, y)$ es determinante para el esquema de multiresolución que presentamos en este capítulo. Al igual que en la §3.6 haremos un estudio de las posibles funciones de pérdida que podemos utilizar basándonos principalmente en los espacios ℓ^p con $p = 1, 2, \ldots, +\infty$. Introducimos también la norma Huber (ver [61]) que combina las normas $p = 1$ y $p = 2$, buscando un equilibrio entre ambas definiéndola como:

$$L_{H,\kappa}(y, g(X)) = \begin{cases} (y - g(X))^2, & \text{si } |y - g(X)| \leq \kappa, \\ 2\kappa(|y - g(X)| - \kappa), & \text{en otro caso.} \end{cases} \tag{4.6}$$

con $g \in \mathcal{K}$, X un vector y κ una constante. En la Figura 4.4 podemos ver que si utilizamos normas $p > 1$ tendremos una fuerte penalización en aquellos valores aproximados que distan mucho de los valores originales. Estos valores se denominan observaciones anómalas.

Consideramos también la función en norma infinito, si tenemos $(X^i, y^i)_{i \in \mathcal{M}_k}$ entonces:

$$L_\infty(y^j, g(X^j)) = \max_{i \in \mathcal{M}_k} |y^i - g(X^i)|, \forall j \in \mathcal{M}_k, g \in \mathcal{K}. \tag{4.7}$$

Esta es la norma Chebyshev que da lugar al problema de aproximación minimax visto en la §3.6. Establecemos una comparativa mostrando un ejemplo sencillo.

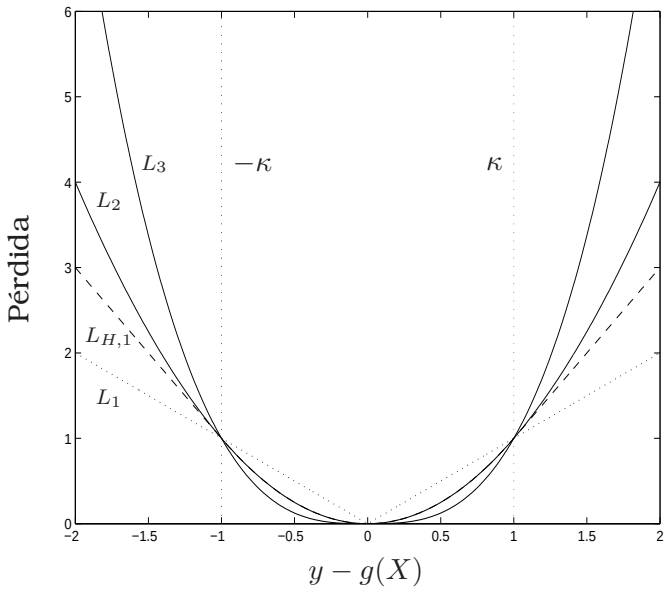


Figura 4.4: *Comparación de las distintas funciones de pérdida,* $L_p(x, y) = |x - y|^p$ *con* $p = 1, 2, 3$ *y la norma Huber* $L_{H,1}$.

Planteamos regresión lineal utilizando los datos de la función $f(x) = x$ en los puntos $i = 1, \ldots, 60$, modificando cuatro puntos $j = 12, 26, 34, 55$ dándole el valor 200, i.e., $f(j) = 200$. Como podemos ver en la Fig. 4.5 la distancia de la aproximación a las observaciones anómalas decrece con p ya que cuando más pequeña es p menos importancia da a dichos valores. En multiresolución nos interesará penalizar errores grandes o

no dependiendo de cuál sea la compresión o el problema que deseemos resolver. Evidentemente cuando queremos hacer una compresión de una señal o una imagen deseamos tener un error pequeño entre la imagen comprimida y la imagen original. Por esto será necesario utilizar norma 1 o Huber con κ cercano a cero como veremos muy claramente en la §4.3.5 en los ejemplos uno dimensionales. Un estudio sobre la robustez de la norma se puede ver también p. ej. en el capítulo 6 de [70], capítulo 10 de [56] o en capítulo 6 de [20] y en [62].

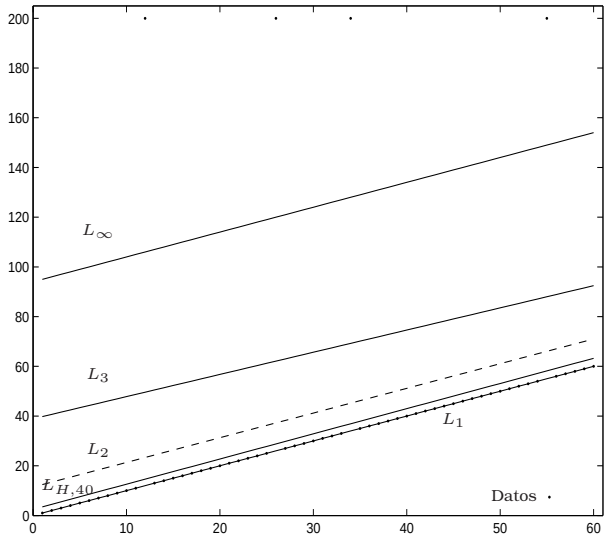


Figura 4.5: *Regresión lineal para el ejemplo* $f(i) = i$ *con* $i \in \{1, \ldots, 60\} \setminus \{12, 26, 34, 55\}$ *y* $f(j) = 200$ *con* $j \in \{12, 26, 34, 55\}$ *utilizando las normas* $L_p(x, y)$ *con* $p = 1, 2, 3, +\infty$ *y la norma Huber* $L_{H,40}(x, y)$*.*

Podemos también utilizar la función de pérdida que penaliza aquellos valores que superan un umbral, κ, llamada *log barrier* que definimos como:

$$L_{log,\kappa}(y, g(X)) = \begin{cases} -\kappa^2 \log\left(1 - \left(\frac{y-g(X)}{\kappa}\right)^2\right), & \text{si } |y - g(X)| \leq \kappa, \\ +\infty & \text{en otro caso,} \end{cases} \tag{4.8}$$

y una función de pérdida que sea 0 en aquellos valores aproximados que no disten de los originales más que un cierto valor, κ, elegido (en la

literatura se le llama función de pérdida *deadzone-linear*) . Así:

$$L_{dea,\kappa}(y, g(X)) = \begin{cases} 0, & \text{si } |y - g(X)| \leq \kappa, \\ |y - g(X)| - \kappa, & \text{en otro caso.} \end{cases} \tag{4.9}$$

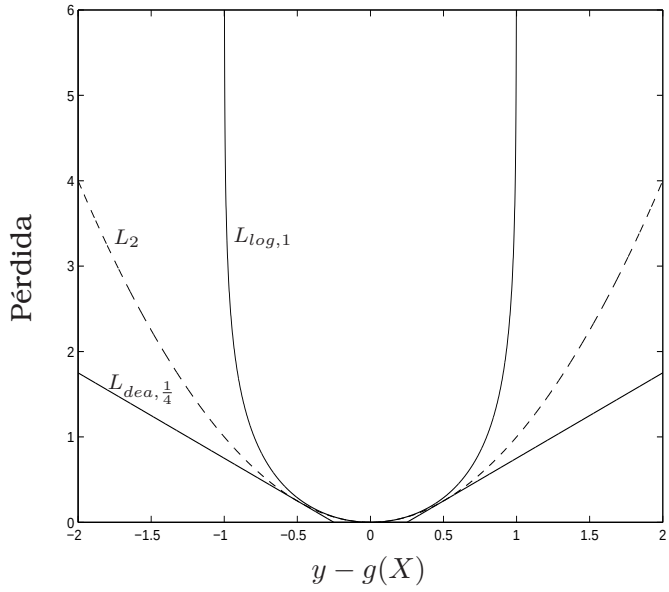


Figura 4.6: *Comparación de las distintas funciones de pérdida:* L_2, $L_{log,1}$ *y* $L_{dea,\frac{1}{4}}$.

Utilizaremos también estas dos funciones en nuestro trabajo, se pueden ver otras funciones como la norma *largest*[2] (ver capítulo 6 de [20]). Para minimizar todas estas normas propondremos la utilización de optimización convexa y dentro de este contexto más particularmente utilizaremos el paquete `cvx` presentado por M. Grant et al. en [49, 50] como en el capítulo 3.

[2]La norma *largest* se basa en cambiar nuestro problema (4.2) eligiendo un valor κ en

$$\arg\min_{g \in \mathcal{K}} \sum_{i=1}^{\kappa} |f^k - g(\mathcal{S}^{r,s}(f^{k-1}))|_{[i]}$$

donde $|f^k - g(\mathcal{S}^{r,s}(f^{k-1}))|_{[1]} \geq |f^k - g(\mathcal{S}^{r,s}(f^{k-1}))|_{[2]} \geq \ldots \geq |f^k - g(\mathcal{S}^{r,s}(f^{k-1}))|_{[|\mathcal{M}|]}$ son los valores $|f_1^k - g(\mathcal{S}^{r,s}(f_1^{k-1}))|, |f_2^k - g(\mathcal{S}^{r,s}(f_2^{k-1}))|, \ldots, |f_{|\mathcal{M}|}^k - g(\mathcal{S}^{r,s}(f_{|\mathcal{M}|}^{k-1}))|$ escogidos en orden decreciente.

Nota 4.3. *En el caso del problema de minimización con función $L_1(x, y)$ podremos utilizar o bien el algoritmo descrito en el capítulo 3 presentado por Vogel [91] y cuya convergencia fue estudiada por T. Chan et al. en [26] o bien el paquete* `cvx` *dependiendo del número de elementos del conjunto de entrenamiento que tengamos. Si el número de elementos del conjunto de entrenamiento es demasiado grande no es conveniente utilizar* `cvx` *pues esta diseñado para sistemas con tamaños pequeños.*

Resumimos y esquematizamos las normas que vamos a utilizar en los ejemplos uno dimensionales:

I. Normas $p \geq 2$. A la hora de calcular el operador predictor da mucha prioridad (dependiendo de p) a errores altos y muy poca o casi nula a errores pequeños. Esto va a provocar que cuando la función a aproximar contenga datos de entrenamiento de diferentes señales, p. ej. una función definida a trozos, halle un predictor más adaptado a la zona donde más error se produzca. Tan solo utilizaremos $p = 2$, el resto de normas tiene un comportamiento similar.

II. Norma *log barrier*. Esta norma en nuestro ejemplos prácticos es bastante similar a la norma 2, incluso da más valor a errores altos.

III. Norma 1. Esta norma es robusta dando la misma prioridad a errores altos que a errores pequeños (solo depende del valor absoluto de cada error). Por tanto se adaptará a cualquier tipo de datos que tengamos. Obtendremos muy buenos resultados uno dimensionales.

IV. Norma *hubber*. Es una norma intermedia entre la norma 1 y la norma 2. El único problema es que trata a los errores pequeños de la misma manera que los trata la norma 2 y a errores grandes de la misma manera que la norma 1 con lo que da prioridad a errores grandes, adaptándose mejor en esas zonas. Apenas toma en cuenta errores pequeños como la norma 2 pero no da un valor tan alto a errores grandes.

V. Norma *deadzone-linear*. Es como la norma 1 pero no toma en cuenta para hallar el funcional aquellos errores que sean menor que una cierta cantidad κ.

En definitiva si deseamos que el operador predictor cometa un error relativamente pequeño en zonas de difícil compresión tenemos que utilizar normas $p > 2$ y *log barrier*. Si por el contrario deseamos que se adapte bien a las zonas donde se cometen errores grandes y que tenga

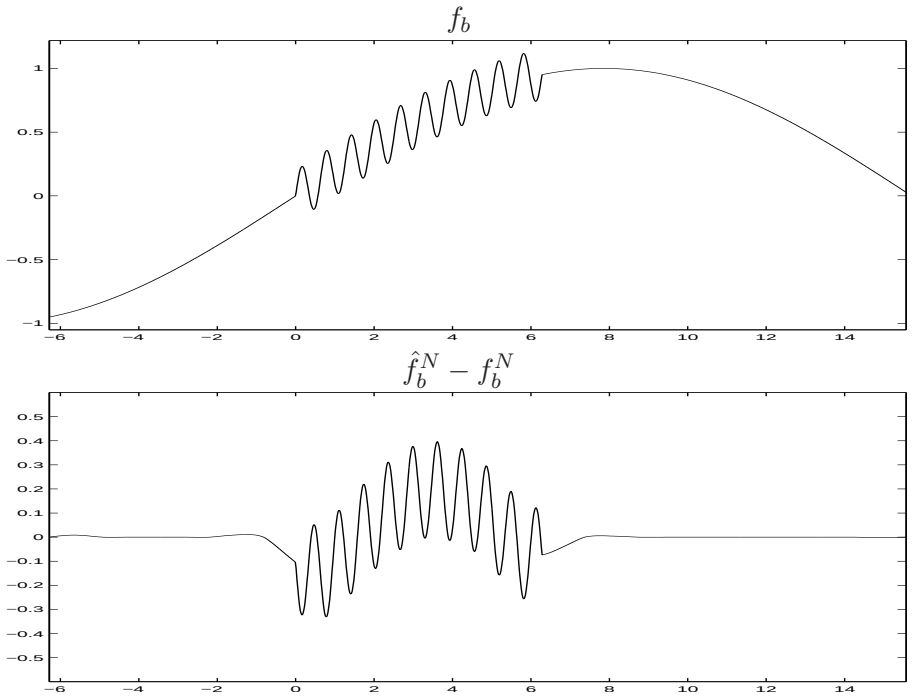
también en cuenta las zonas donde se cometen errores pequeños tendríamos que utilizar norma *hubber*. Si deseamos que se adapte a todas las zonas dependiendo únicamente del error que se cometa en valor absoluto entonces utilizaremos la norma 1.

Veamos estas diferencias mediante un ejemplo. Tomando la función:

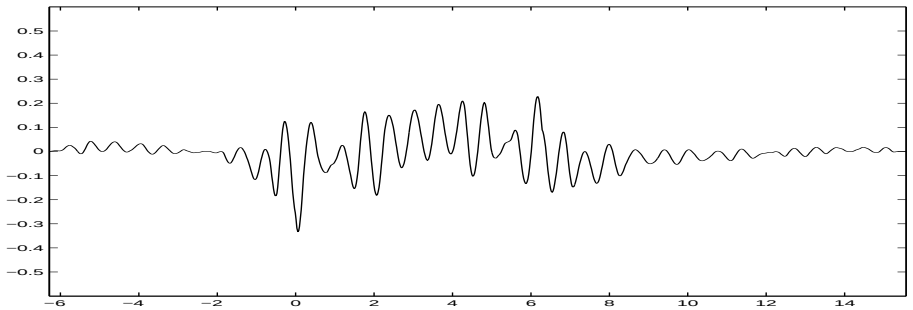
$$f_b(x) = \begin{cases} \sin(0{,}2x) + 0{,}2\sin(10x), & x \in [0, 2\pi[; \\ \sin(0{,}2x), & \text{en otro caso,} \end{cases}$$

discretizada en el contexto de valores puntuales en el intervalo $[-2\pi, 4\pi + 3[$ con $N = 10$. Aplicamos el algoritmo de transformación directa con $L = 6$ (i.e. $N_0 = 4$) hallando los operadores predicción en cada paso con las distintas funciones de pérdida vistas en esta sección. Truncamos todos los detalles a 0 y aplicamos el algoritmo de reconstrucción sin detalles. Comparamos la reconstrucción obtenida $\hat{f}_b$ con la función original f_b obteniendo las Figuras 4.7 y 4.8. El método PV_4 es el método de interpolación segmentaria centrado con cuatro nodos. Podemos ver que este método produce muy poco error en las zonas con poca oscilación y sin embargo produce grandes errores en las zonas con oscilación alta. Si utilizamos el predictor basado en la norma L_2, ésta se ocupa de minimizar la parte con mayor error (la parte muy oscilatoria) "*olvidándose*" del resto y cometiendo en esas zonas errores como podemos ver en la Figura 4.7-(b). Sin embargo si utilizamos norma 1 todos los errores tienen la misma importancia dependiendo únicamente de su módulo. Así, como observamos en 4.7-(c) tenemos en la parte oscilatoria menos errores que PV pero más que utilizando la norma 2 y en el resto más error que utilizando PV pero menos que utilizando la norma 2. Utilizando las funciones de pérdida L_{hub} y L_{dea} dependiendo del valor κ nos acercamos más a la función L_2 o a la función L_1.

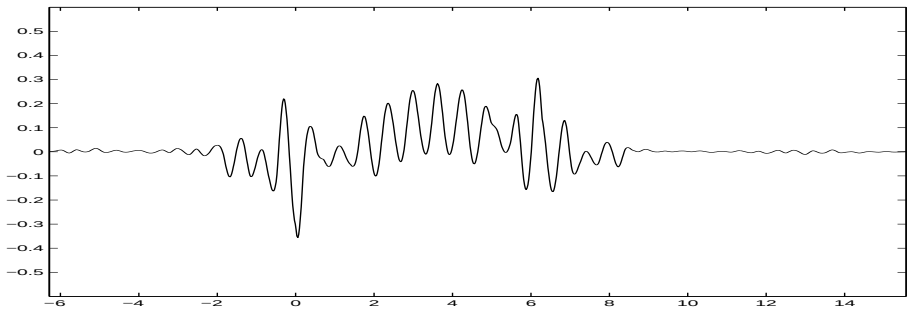
Donde se produce un mayor error es en $x = 0$ y en $x = 2\pi$ donde la función no es diferenciable. La función de pérdida L_∞ se va a preocupar tan sólo de no cometer un error alto en cada punto. Así pues como vemos en 4.8-(c) no comete un error demasiado grande en ningún punto. Sin embargo al minimizar los errores grandes para no obtener un error máximo demasiado elevado comete demasiado error en el resto de puntos donde los errores son más pequeños y por tanto menores que el error máximo.



(a) PV_4, $E_2 = 0{,}1024$

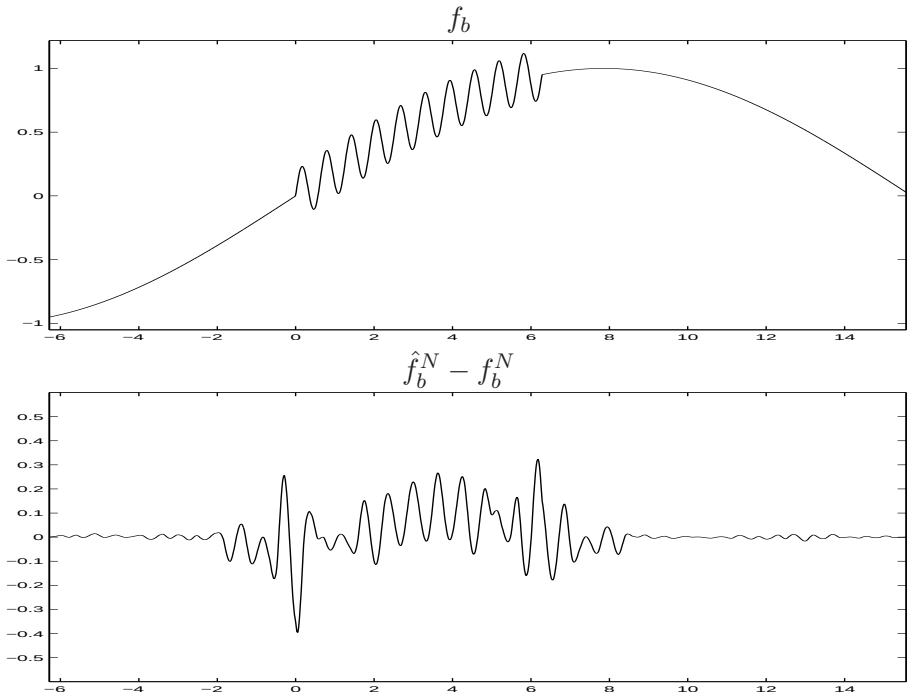


(b) ${}^2\mathrm{LM}_4$, $E_2 = 0{,}0714$

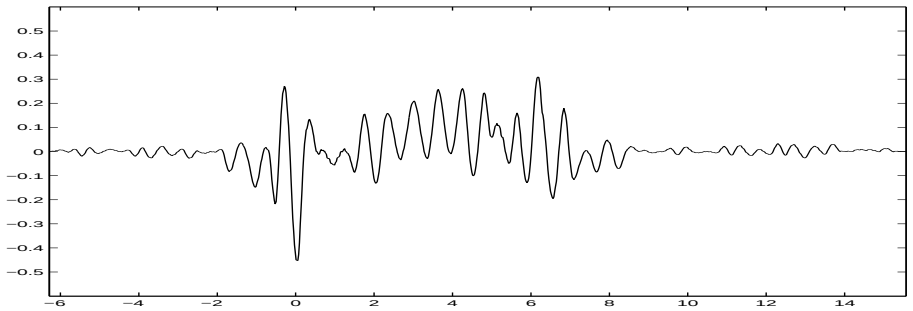


(c) ${}^1\mathrm{LM}_4$, $E_2 = 0{,}0801$

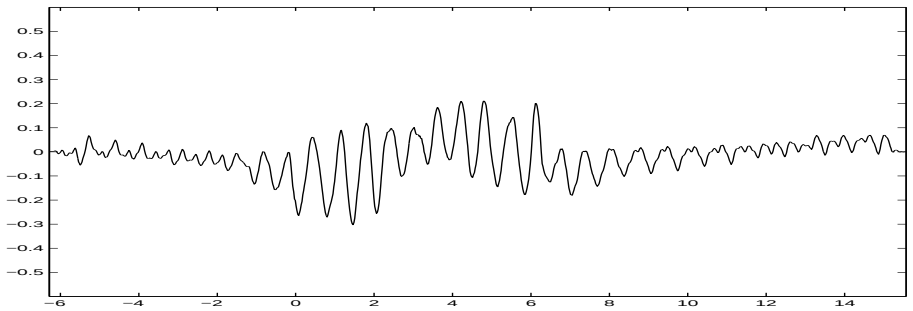
Figura 4.7: *Error entre la función original f_b y la aproximación después de aplicar los algoritmos de multiresolución sin almacenar ningún detalle utilizando los métodos: (a) PV_4, (b) ${}^2LM_4$ y (c) ${}^1LM_4$.*



(a) $^{hub,0,25}\mathbf{LM}_4$, $E_2 = 0{,}0714$



(b) $^{dea,0,25}\mathbf{LM}_4$, $E_2 = 0{,}0707$



(c) $^{\infty}\mathbf{LM}_4$, $E_2 = 0{,}0818$

Figura 4.8: *Error entre la función original f_b y la aproximación después de aplicar los algoritmos de multiresolución sin almacenar ningún detalle utilizando los métodos: (a) $^{hub,0,25}LM_4$, (b) $^{dea,0,25}LM_4$ y (c) $^{\infty}LM_4$.*

4.2.3 Experimentos numéricos

Utilizaremos la discretización basada en valores puntuales (ver §2.2.2) para los ejemplos uno dimensionales.

Como tenemos control del error aplicamos directamente el algoritmo de multiresolución obteniendo:

$$(f^0, \hat{\mathcal{P}}_0^1, d^1, \ldots, \hat{\mathcal{P}}_{N-1}^N, d^N) \tag{4.10}$$

y truncamos los detalles:

$$\hat{d}_j^k = \begin{cases} d_j^k, & \textbf{si } |d_j^k| \geq \varepsilon_k; \\ 0, & \textbf{si } |d_j^k| < \varepsilon_k. \end{cases}$$

con $j = 0, \ldots, J_k$ obteniendo:

$$(f^0, \hat{\mathcal{P}}_0^1, \hat{d}^1, \ldots, \hat{\mathcal{P}}_{N-1}^N, \hat{d}^N). \tag{4.11}$$

No cambiamos los operadores predicción en cada nivel. También utilizaremos la técnica de control del error presentada por Aràndiga et al. en [2] mencionada en el capítulo 3 hallando un operador predictor adaptado a los datos que obtenemos en cada nivel de resolución. A este método lo denotaremos añadiendo "-ec".

Mediremos el error que cometemos en las distintas normas E_1, E_2, E_∞ vistas en los capítulos 2 y 3. Veremos el nivel de compresión contabilizando la cantidad de detalles, NNZ, que almacenamos. En el caso de LM indicaremos con una suma aquellos elementos que se guardan por ser la predicción en cada nivel.

Denotaremos los métodos LM como ${}^p\text{LM}_m$, siendo m el número de puntos utilizado (siempre centrado) y p la función de pérdida utilizada. Partiremos de un nivel de discretización $N = 10$ hasta $N_0 = 4$. Por tanto tenemos que guardar $(N - N_0)m = 6m$ elementos correspondientes a los operadores predicción.

Tomaremos diferentes funciones intentando ilustrar lo visto hasta ahora en el capítulo. La primera función que vamos a tomar es

$$f_d(x) = 40(x + 1/4)^2 \sin(40(x + 1/2))$$

discretizada en el intervalo $[0, 1]$, Figura 4.9.

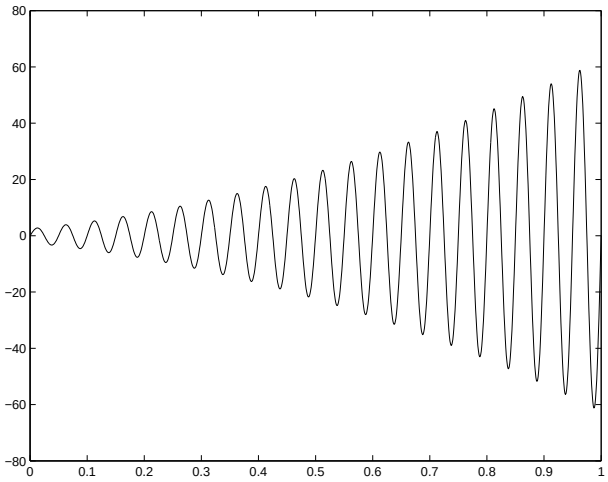


Figura 4.9: *Función f_d.*

Esta función es especialmente complicada de comprimir con métodos lineales (PV) como vemos en la Tabla 4.1. En los métodos LM tenemos una gran capacidad de compresión ya que el operador predicción tiene mucha información. No tenemos grandes diferencias entre los métodos utilizando L_1 y L_2 ya que las diferencias entre las aproximaciones y los valores reales son muy altas e iguales en todos los datos de entrenamiento. En el siguiente ejemplo veremos como si hay una zona oscilatoria y otra zona no oscilatoria la norma 2 minimiza los errores cometidos en la zona oscilatoria "olvidándose" del resto y así obteniendo peores resultados.

	$m = 4, \quad \varepsilon = 10^{-3}$			
	E_1	E_2	E_∞	NNZ
PV_4	$1{,}16 \cdot 10^{-4}$	$2{,}66 \cdot 10^{-4}$	$9{,}97 \cdot 10^{-4}$	741
${}^{2}\mathrm{LM}_4$	$1{,}83 \cdot 10^{-4}$	$2{,}95 \cdot 10^{-4}$	$9{,}97 \cdot 10^{-4}$	$79 + 24$
${}^{1}\mathrm{LM}_4$	$1{,}80 \cdot 10^{-4}$	$2{,}78 \cdot 10^{-4}$	$9{,}40 \cdot 10^{-4}$	$66 + 24$
${}^{hub,10^{-1}}\mathrm{LM}_4$	$1{,}83 \cdot 10^{-4}$	$2{,}95 \cdot 10^{-4}$	$9{,}97 \cdot 10^{-4}$	$79 + 24$
${}^{dea,10^{-3}}\mathrm{LM}_4$	$3{,}11 \cdot 10^{-4}$	$4{,}91 \cdot 10^{-4}$	$2{,}41 \cdot 10^{-3}$	$75 + 24$
${}^{log,10^{-3}}\mathrm{LM}_4$	$1{,}83 \cdot 10^{-4}$	$2{,}95 \cdot 10^{-4}$	$9{,}98 \cdot 10^{-4}$	$79 + 24$
${}^{\infty}\mathrm{LM}_4$	$1{,}20 \cdot 10^{-4}$	$2{,}12 \cdot 10^{-4}$	$1{,}20 \cdot 10^{-3}$	$95 + 24$

Tabla 4.1: *Función f_d. Errores y detalles almacenados utilizando métodos LM con 4 puntos.*

Podemos ver que la utilización de otras normas en este caso es irrelevante pues los resultados son similares a L_1 y L_2. Por esto en este ejemplo tan solo vamos a centrarnos en estas dos normas. Vamos a hacer dos experimentos: el primero reduciendo el valor de ε hasta obtener una función reconstruida "casi" sin pérdida; el segundo dando un valor muy alto a ε para no guardar casi ningún detalle utilizando los métodos LM. Obtenemos la Tabla 4.2.

	$m = 4, \quad \varepsilon = 10^{-5}$			
	E_1	E_2	E_∞	NNZ
PV_4	$1{,}66 \cdot 10^{-8}$	$2{,}96 \cdot 10^{-7}$	$7{,}78 \cdot 10^{-6}$	1004
${}^2LM_4$	$1{,}71 \cdot 10^{-6}$	$2{,}56 \cdot 10^{-6}$	$9{,}62 \cdot 10^{-6}$	$216 + 24$
${}^2LM_4$-ec	$1{,}70 \cdot 10^{-6}$	$2{,}55 \cdot 10^{-6}$	$9{,}55 \cdot 10^{-6}$	$216 + 24$
${}^1LM_4$	$2{,}27 \cdot 10^{-6}$	$3{,}42 \cdot 10^{-6}$	$1{,}47 \cdot 10^{-5}$	$344 + 24$
${}^1LM_4$-ec	$1{,}78 \cdot 10^{-6}$	$2{,}70 \cdot 10^{-6}$	$9{,}98 \cdot 10^{-6}$	$202 + 24$
	$\varepsilon = 1$			
PV_4	0,2199	0,3176	1,04	105
${}^2LM_4$	0,0188	0,0780	0,90	$6 + 24$
${}^1LM_4$	0,0184	0,0781	0,90	$6 + 24$

Tabla 4.2: *Función f_d. Errores y detalles almacenados utilizando métodos LM con 4 puntos.*

Vemos que con poca cantidad de detalles tenemos una buena aproximación utilizando los métodos LM. En el caso PV hay que guardar casi la misma cantidad de elementos que tiene la función original. En el caso que $\varepsilon = 1$ vemos que obtenemos normas $1, 2, \infty$ mucho menores guardando la tercera parte de valores.

Hemos elegido este ejemplo porque todo el conjunto de entrenamiento es bastante similar. Hacemos un segundo ejemplo donde los datos de entrenamiento cambian, veamos como ahora la norma 2 necesita una gran cantidad de detalles para obtener una buena aproximación. Tomamos la función mencionada en la sección anterior:

$$f_b(x) = \begin{cases} \sin(0{,}2x) + 0{,}2\sin(10x), & x \in [0, 2\pi[; \\ \sin(0{,}2x), & \text{en otro caso} \end{cases}$$

discretizada en el intervalo $[-2\pi, 4\pi + 3[$, Figura (a)-4.7.

En este caso como vemos en la Tabla 4.3 el método ${}^2LM_4$ pierde su capacidad de compresión necesitando muchos detalles para obtener una aproximación menor que $\varepsilon = 10^{-3}$. Sin embargo con ${}^1LM_4$ guardando aproximadamente la tercera parte de los detalles que almacenamos al

utilizar el método lineal PV obtenemos menos error. También obtenemos buenos resultados para el método ${}^{dea,10^{-3}}\mathrm{LM}_4$.

	$m=4, \quad \varepsilon=10^{-3}$			
	E_1	E_2	E_∞	NNZ
PV_4	$6{,}17\cdot10^{-5}$	$1{,}27\cdot10^{-4}$	$9{,}30\cdot10^{-4}$	147
${}^{2}\mathrm{LM}_4$	$2{,}27\cdot10^{-4}$	$3{,}40\cdot10^{-4}$	$1{,}21\cdot10^{-3}$	$185+24$
${}^{1}\mathrm{LM}_4$	$1{,}19\cdot10^{-5}$	$5{,}81\cdot10^{-5}$	$5{,}48\cdot10^{-4}$	**47+24**
${}^{hub,10^{-1}}\mathrm{LM}_4$	$2{,}26\cdot10^{-4}$	$3{,}22\cdot10^{-4}$	$1{,}10\cdot10^{-3}$	$190+24$
${}^{dea,10^{-3}}\mathrm{LM}_4$	$7{,}39\cdot10^{-4}$	$9{,}81\cdot10^{-4}$	$3{,}71\cdot10^{-3}$	**66+24**
${}^{log,10^{-3}}\mathrm{LM}_4$	$2{,}26\cdot10^{-4}$	$3{,}38\cdot10^{-4}$	$1{,}21\cdot10^{-3}$	$186+24$
${}^{\infty}\mathrm{LM}_4$	$9{,}51\cdot10^{-5}$	$1{,}76\cdot10^{-4}$	$8{,}95\cdot10^{-4}$	$621+24$

Tabla 4.3: *Función f_b. Errores y detalles almacenados utilizando métodos LM con 4 puntos.*

Añadamos ahora dos partes oscilatorias y una parte no oscilatoria utilizando la función:

$$f_e(x)=\begin{cases} 40\sin(40\pi x), & 0\le x\le 7/20;\\ -1{,}96, & 7/20\le x\le 267/500;\\ 10\sin(30\pi x)-2, & 267/500\le x\le 1,\end{cases}$$

discretizada en el intervalo $[0,1]$, Figura 4.10.

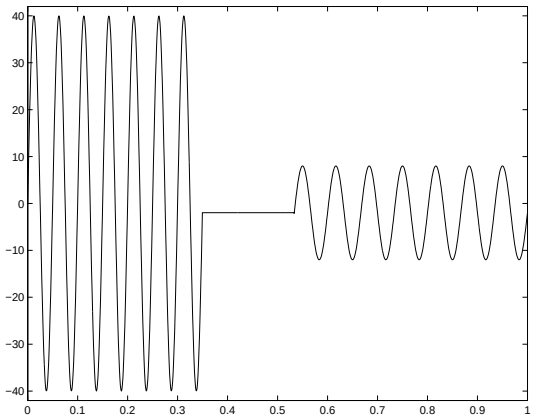


Figura 4.10: *Función f_e.*

En este caso obtenemos más NNZ utilizando los métodos ${}^{2,hub,\infty}\mathrm{LM}_4$. Se obtienen buenos resultados con función de pérdida L_1 reduciendo a

más de la mitad la cantidad de detalles para almacenar y aminorando el error en norma $1, 2, \infty$.

	$m = 4, \quad \varepsilon = 10^{-3}$			
	E_1	E_2	E_∞	NNZ
PV_4	$7{,}36 \cdot 10^{-5}$	$1{,}72 \cdot 10^{-4}$	$9{,}83 \cdot 10^{-4}$	554
${}^{2}\mathrm{LM}_4$	$8{,}71 \cdot 10^{-6}$	$7{,}40 \cdot 10^{-5}$	$8{,}56 \cdot 10^{-4}$	$991 + 24$
${}^{1}\mathrm{LM}_4$	$1{,}49 \cdot 10^{-4}$	$2{,}97 \cdot 10^{-4}$	$9{,}67 \cdot 10^{-4}$	**226+24**
${}^{hub,10^{-1}}\mathrm{LM}_4$	$1{,}97 \cdot 10^{-4}$	$3{,}39 \cdot 10^{-4}$	$1{,}32 \cdot 10^{-3}$	$829 + 24$
${}^{dea,10^{-3}}\mathrm{LM}_4$	$5{,}12 \cdot 10^{-4}$	$7{,}99 \cdot 10^{-4}$	$3{,}24 \cdot 10^{-3}$	**336+24**
${}^{log,10^{-3}}\mathrm{LM}_4$	$8{,}71 \cdot 10^{-6}$	$7{,}40 \cdot 10^{-5}$	$8{,}56 \cdot 10^{-4}$	$991 + 24$
${}^{\infty}\mathrm{LM}_4$	$3{,}48 \cdot 10^{-6}$	$4{,}82 \cdot 10^{-5}$	$9{,}01 \cdot 10^{-4}$	$1002 + 24$

Tabla 4.4: *Función f_e. Errores y detalles almacenados utilizando métodos LM con 4 puntos.*

Como último experimento vemos una función que tiene una discontinuidad, una zona suave y una zona oscilatoria, así:

$$f_a(x) = \begin{cases} \sin(0{,}2x), & x \in [-2\pi, 0[\cup[2\pi, 4\pi[; \\ \sin(0{,}2x) + 0{,}2\sin(10x), & x \in [0, 2\pi[; \\ \sin(0{,}2x) + 1, & \text{en otro caso.} \end{cases}$$

discretizada en el intervalo $[-2\pi, 4\pi + 3[$, Figura 4.11.

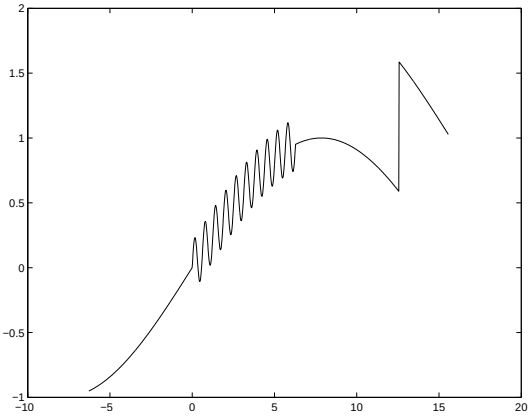


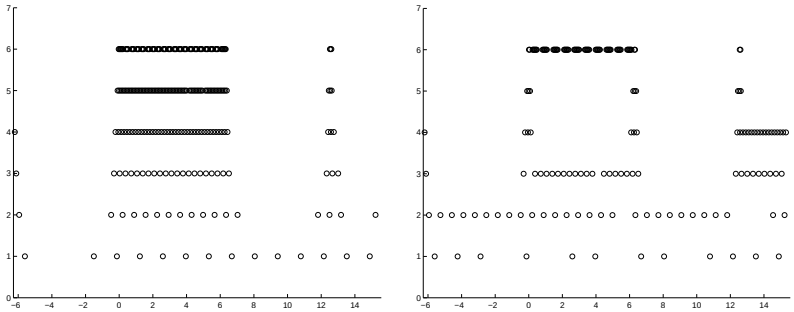
Figura 4.11: *Función f_a.*

En la Tabla 4.5 podemos ver que los datos obtenidos para ${}^{1}\mathrm{LM}_4$ son realmente buenos teniendo en cuenta la dificultad de compresión de la

señal f_a. El resto de métodos no supera, incluso empeora, los resultados obtenidos por el método lineal.

	$m = 4, \quad \varepsilon = 10^{-3}$			
	E_1	E_2	E_∞	NNZ
PV_4	$5{,}37 \cdot 10^{-5}$	$1{,}24 \cdot 10^{-4}$	$9{,}30 \cdot 10^{-4}$	165
${}^{2}\mathrm{LM}_4$	$1{,}97 \cdot 10^{-4}$	$3{,}71 \cdot 10^{-4}$	$1{,}55 \cdot 10^{-3}$	$670 + 24$
${}^{1}\mathrm{LM}_4$	$6{,}66 \cdot 10^{-5}$	$1{,}20 \cdot 10^{-4}$	$5{,}47 \cdot 10^{-4}$	**89+24**
${}^{hub,10^{-1}}\mathrm{LM}_4$	$2{,}67 \cdot 10^{-4}$	$4{,}50 \cdot 10^{-4}$	$1{,}96 \cdot 10^{-3}$	$576 + 24$
${}^{dea,10^{-3}}\mathrm{LM}_4$	$5{,}38 \cdot 10^{-4}$	$6{,}82 \cdot 10^{-4}$	$1{,}82 \cdot 10^{-3}$	$131 + 24$
${}^{log,10^{-3}}\mathrm{LM}_4$	$1{,}97 \cdot 10^{-4}$	$3{,}71 \cdot 10^{-4}$	$1{,}55 \cdot 10^{-3}$	$670 + 24$
${}^{\infty}\mathrm{LM}_4$	$1{,}20 \cdot 10^{-4}$	$2{,}88 \cdot 10^{-4}$	$1{,}55 \cdot 10^{-3}$	$832 + 24$

Tabla 4.5: *Función f_a. Errores y detalles almacenados utilizando métodos LM con 4 puntos.*



(a) PV_4 (b) ${}^{1}\mathrm{LM}_4$

Figura 4.12: *Función f_a. Detalles almacenados: (a) PV_4 y (b) ${}^{1}LM_4$.*

En conclusión los operadores predicción hallados utilizando la función L_2 reducen los errores en las zonas de difícil compresión como son zonas oscilatorias. El problema se produce cuando la función presenta zonas oscilatorias y no oscilatorias entonces el método ${}^{2}\mathrm{LM}$ pierde su eficacia. Para solucionarlo hemos introducido otras normas más robustas obteniendo buenos resultados con L_1 en todo tipo de señales.

4.3 Multiresolución de aprendizaje 2D

Del mismo modo que hemos hecho en una dimensión podemos definir la multiresolución de aprendizaje en dos dimensiones. La única diferencia que tendremos será la elección del *stencil*, en el caso dos dimensional será una ventana cuadrada (por no complicar la notación). Así si denotamos como $f^k_{i,j} = (\mathcal{D}_k f)_{i,j}$ con $i, j \in \mathcal{M} \subset \mathbb{N}$ con $\mathcal{D}_k$ un operador discretización (ver capítulo 2) y $r,\ s \in \mathbb{N}$ tenemos que:

$$\mathcal{S}^{r,s}(f^k_{i,j}) = (f^k_{i-r,j-r}, \ldots, f^k_{i+s,j+s}),\ i, j \in \mathcal{M},$$

que son los valores de la discretización en los valores próximos a $f^k_{i,j}$.

Nota 4.4. *Lo explicamos en el contexto de medias en celda, sería análogo para cualquier otro operador discretización.*

El problema de aprendizaje, como ya hemos visto en una dimensión viene determinado por minimizar el funcional:

$$\hat{\mathcal{P}}^k_{k-1} = \underset{g\in\mathcal{K}}{\arg\min} \sum_{i,j\in\mathcal{M}} L(f^k_{2i,2j}, g(\mathcal{S}^{r,s}(f^{k-1}_{i,j}))). \tag{4.12}$$

Veamos un ejemplo en el contexto de medias en celda.

Ejemplo 4.3. *Definimos el operador predicción para medias en celda en dos dimensiones. Sabiendo que el operador decimación es*

$$(\mathcal{D}^{k-1}_k f^{k-1})_{i,j} = \frac{1}{4}(f^k_{2i,2j} + f^k_{2i-1,2j} + f^k_{2i,2j-1} + f^k_{2i-1,2j-1}) \quad 1 \le i, j \le J_k.$$

Entonces definimos tres operadores predicción uno para cada píxel $(2i, 2j)$, $(2i-1, 2j)$, $(2i, 2j-1)$ *minimizando los siguientes funcionales:*

$$(\hat{\mathcal{P}}_{0,0})^k_{k-1} = \underset{g\in\mathcal{K}}{\arg\min} \sum_{i,j=r+1}^{J_{k-1}-s} L(f^k_{2i,2j}, g(\mathcal{S}^{r,s}(f^{k-1}_{i,j}))),$$

$$(\hat{\mathcal{P}}_{0,-1})^k_{k-1} = \underset{g\in\mathcal{K}}{\arg\min} \sum_{i,j=r+1}^{J_{k-1}-s} L(f^k_{2i,2j-1}, g(\mathcal{S}^{r,s}(f^{k-1}_{i,j}))),$$

$$(\hat{\mathcal{P}}_{-1,0})^k_{k-1} = \underset{g\in\mathcal{K}}{\arg\min} \sum_{i,j=r+1}^{J_{k-1}-s} L(f^k_{2i-1,2j}, g(\mathcal{S}^{r,s}(f^{k-1}_{i,j}))).$$

Y definimos el operador predicción utilizando la estrategia (E0) vista en la §2.3 como:

$$\begin{aligned}
(\mathcal{P}^k_{k-1}f^{k-1})_{2i,2j} &= (\hat{\mathcal{P}}_{0,0})^k_{k-1}(\mathcal{S}^{r,s}(f^{k-1}_{i,j})), \quad r+1 \le i,j \le J_{k-1}-s;\\
(\mathcal{P}^k_{k-1}f^{k-1})_{2i,2j-1} &= (\hat{\mathcal{P}}_{0,-1})^k_{k-1}(\mathcal{S}^{r,s}(f^{k-1}_{i,j})), \quad r+1 \le i,j \le J_{k-1}-s;\\
(\mathcal{P}^k_{k-1}f^{k-1})_{2i-1,2j} &= (\hat{\mathcal{P}}_{-1,0})^k_{k-1}(\mathcal{S}^{r,s}(f^{k-1}_{i,j})), \quad r+1 \le i,j \le J_{k-1}-s;\\
(\mathcal{P}^k_{k-1}f^{k-1})_{2i-1,2j-1} &= 4f^{k-1}_{i,j} - (\hat{\mathcal{P}}_{0,0})^k_{k-1}(\mathcal{S}^{r,s}(f^{k-1}_{i,j})) - (\hat{\mathcal{P}}_{0,-1})^k_{k-1}(\mathcal{S}^{r,s}(f^{k-1}_{i,j}))-\\
&\quad - (\hat{\mathcal{P}}_{-1,0})^k_{k-1}(\mathcal{S}^{r,s}(f^{k-1}_{i,j})), \quad r+1 \le i,j \le J_{k-1}-s.
\end{aligned}$$

Podemos ver una variante a esta definición utilizando la estrategia (AY) vista en la §2.3 introduciendo un funcional más para el píxel $(2i-1, 2j-1)$.

Los puntos en la frontera los hallaremos utilizando un operador predicción basado en interpolación polinómica segmentaria descentrado.

4.3.1 Primer problema: paso por discontinuidades. Posible solución: *Learning-based multiresolution edge-dependent*

En dos dimensiones el conjunto de entrenamiento es mucho mayor que en una dimensión, esto provoca que el operador predicción se adapte menos a las discontinuidades. Por ejemplo en una imagen real como lena, la gran mayoría de valores de entrenamiento no pertenecen a un contorno y por tanto la predicción que hallemos no estará adaptada a éstos, tenderá a la predicción basada en interpolación.

Este problema plantea una cuestión: *¿cómo podemos adaptar nuestro operador a los contornos de una imagen?* Para solucionarlo hacemos una clasificación de los píxeles de la imagen en cinco conjuntos: orientación horizontal, vertical, diagonal positiva, diagonal negativa y puntos que marcamos como constantes. Para eso utilizaremos el detector de contornos Sobel, ver [81]. También podríamos utilizar otros detectores (Canny [24], Prewitt [77]) y otras clasificaciones (ver p. ej. [76, 89]). Con esta clasificación planteamos un problema similar a (4.12) para cada orientación.

Así pues el algoritmo de multiresolución tendría dos pasos:

1. Detección de contornos: Consiste en tomar los puntos en el nivel $k-1$ y dividirlos en cinco conjuntos dependiendo de la orientación que nos marque el detector de contornos. Denotaremos como Γ^{k-1}

los puntos que no nos marque como contorno. Dentro de los puntos que nos marca como contorno tendremos $\Gamma_{\downarrow}^{k-1}$ puntos verticales, $\Gamma_{\rightarrow}^{k-1}$ puntos horizontales, $\Gamma_{\searrow}^{k-1}$ puntos diagonal positivos, $\Gamma_{\swarrow}^{k-1}$ puntos diagonal negativos.

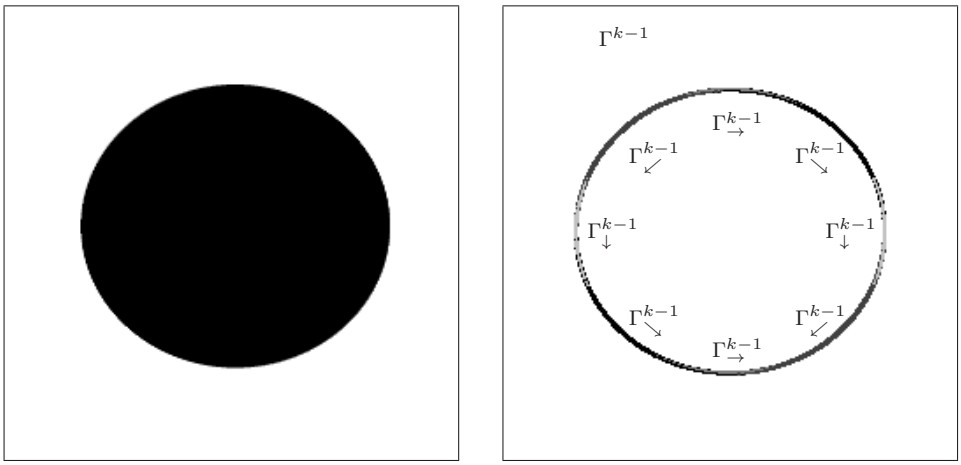
2. Realizada esta clasificación de los puntos hallamos un operador predicción para cada conjunto, es decir dividimos nuestro problema (4.2) en cinco problemas similares:

$$
\begin{aligned}
(\hat{\mathcal{P}}_{\rightarrow})_{k-1}^{k} &= \arg\min_{g\in\mathcal{K}} \sum_{i,j\in\Gamma_{\rightarrow}^{k-1}} L(f_{2i,2j}^{k}, g(\mathcal{S}^{r,s}(f_{i,j}^{k-1}))),\\
(\hat{\mathcal{P}}_{\downarrow})_{k-1}^{k} &= \arg\min_{g\in\mathcal{K}} \sum_{i,j\in\Gamma_{\downarrow}^{k-1}} L(f_{2i,2j}^{k}, g(\mathcal{S}^{r,s}(f_{i,j}^{k-1}))),\\
(\hat{\mathcal{P}}_{\searrow})_{k-1}^{k} &= \arg\min_{g\in\mathcal{K}} \sum_{i,j\in\Gamma_{\searrow}^{k-1}} L(f_{2i,2j}^{k}, g(\mathcal{S}^{r,s}(f_{i,j}^{k-1}))),\\
(\hat{\mathcal{P}}_{\swarrow})_{k-1}^{k} &= \arg\min_{g\in\mathcal{K}} \sum_{i,j\in\Gamma_{\swarrow}^{k-1}} L(f_{2i,2j}^{k}, g(\mathcal{S}^{r,s}(f_{i,j}^{k-1}))),\\
\hat{\mathcal{P}}_{k-1}^{k} &= \arg\min_{g\in\mathcal{K}} \sum_{i,j\in\Gamma^{k-1}} L(f_{i,j}^{k}, g(\mathcal{S}^{r,s}(f_{i,j}^{k-1}))).
\end{aligned}
$$

Veamos un ejemplo donde queda reflejada esta división de puntos que utilizaremos como conjunto de entrenamiento para cada uno de los operadores predicción.

Nota 4.5. *Es evidente que algunos píxeles marcados como horizontales podrían ser "mal" marcados dependiendo del detector de contornos que utilicemos. En aplicaciones en imágenes que un conjunto pequeño de puntos esté mal marcado no afecta demasiado al planteamiento y solución del funcional ya que el conjunto de entrenamiento es muy grande.*

Nota 4.6. *En imágenes podríamos hacer una clasificación con mayor número de orientaciones (ver [89]) o incluso introduciendo un nuevo operador predicción para aquellos puntos que hacen esquina (en la literatura llamados* corner points*, para su detección ver p. ej. [58, 68]) pero en ambos casos no produce un gran beneficio, es decir menos detalles para almacenar, sino que además suponen más coste de almacenamiento al tener que guardar los filtros obtenidos para estos puntos.*



(a) (b)

Figura 4.13: *(a) Imagen original; (b) División de los píxeles de los contornos según su orientación.*

Las principales ventajas que presenta el método adaptado a contornos son:

1. Aproxima muy bien en las zonas suaves ya que el predictor no se ve alterado por posibles discontinuidades.

2. Permite una mayor compresión pues los filtros se adaptan a los contornos.

3. Mayor calidad visual de la imagen resultante, se distinguen mejor los contornos.

Sin embargo no todo son ventajas, el método computacionalmente es más lento en la ejecución del algoritmo. Ya no tenemos control del error ya que al cuantizar (o truncar) los detalles algunos puntos que estaban marcados con una determinada orientación pueden cambiarla. Por último **mayor coste en el almacenamiento** de los filtros ya que tenemos uno para cada orientación. Por ello planteamos una alternativa en la siguiente sección.

4.3.2

Segundo problema: coste de almacenamiento del operador predictor. Posible solución: *Learning-based multiresolution non-level-dependent*

Dos son los principales problemas que plantea el almacenamiento de los filtros, en esta sección proponemos soluciones.

Representación digital

Al tener que almacenar el(los) operador(es) predictor debemos tener en cuenta su representación digital, es decir, no es lo mismo almacenar (computacionalmente hablando) un valor real que un valor entero. Elegiremos un valor 2^μ, con $\mu \in \mathbb{N}$, recordemos que μ bits almacenan hasta el valor $2^\mu - 1$; en una imagen $\mu = 8$. Supongamos que los valores hallados en el nivel $k-1$ son $\{a^j_{\hat{\mathcal{P}}^k_{k-1}}\}_{j\in\mathcal{M}}$ correspondientes al predictor $\hat{\mathcal{P}}^k_{k-1}$. Así almacenaremos:

$$\hat{a}^j_{\hat{\mathcal{P}}^k_{k-1}} = [2^\mu a^j_{\hat{\mathcal{P}}^k_{k-1}}], \quad j \in \mathcal{M}_k; \tag{4.13}$$

donde $[\cdot]$ es el entero obtenido por redondeo. Si los valores $a^j_{\hat{\mathcal{P}}^k_{k-1}} \in [-2^\lambda, 2^\lambda]$ entonces $\hat{a}^j_{\hat{\mathcal{P}}^k_{k-1}} \in [-2^{\lambda+\mu}, 2^{\lambda+\mu}]\ \forall\, j$ y por tanto utilizaremos $\lambda + \mu$ bits para cada valor (en nuestro caso tendremos $\mu = 8$ y $\lambda \geq 0$). En el algoritmo de reconstrucción recuperaremos una aproximación de los filtros originales multiplicando los almacenados por $2^{-\mu}$. En los ejemplos numéricos que mostramos en la §4.3.5 solo mediremos cuantos valores debemos almacenar sin expresar su coste de almacenamiento en número de bits.

Learning-based multiresolution non-level-dependent

En el esquema de multiresolución presentado utilizando técnicas estadísticas de aprendizaje teníamos que

$$f^N \equiv (f^0, \hat{\mathcal{P}}^1_0, d^1, \hat{\mathcal{P}}^2_1, d^2, \ldots, \hat{\mathcal{P}}^N_{N-1}, d^N)$$

por tanto debemos almacenar los predictores en cada uno de los niveles de la multiresolución. Así pues planteamos una alternativa introduciendo como conjunto de entrenamiento todos los posibles valores en todos

los niveles de multiresolución, i. e. tendríamos $\{(\mathcal{S}^{r,s}(f_{i,j}^{k-1}), f_{2i,2j}^{k})_{i,j\in\mathcal{M}_k}\}_{k=1}^{N}$. El problema quedaría como:

$$\hat{\mathcal{P}}_f = \arg\min_{g\in\mathcal{K}} \sum_{k=1}^{N} \sum_{i,j\in\mathcal{M}_k} L(f_{2i,2j}^{k}, g(\mathcal{S}^{r,s}(f_{i,j}^{k-1}))),$$

por tanto tan solo deberíamos almacenar un único filtro para todos los niveles, a esta modificación en el esquema le llamaremos *Learning-based multiresolution non-level-dependent*[3]: LM-nld.

Este método nos sugiere la pregunta: *¿existe un operador predicción que sea bueno para un conjunto (finito) de señales o imágenes determinado con independencia del nivel (con dependencia de la función de pérdida, $L(x,y)$ y del conjunto de funciones donde minimizar, $\mathcal{K}$)?*.

Plantearíamos por tanto el siguiente problema: Sea $\mathcal{Z}$ un conjunto de señales o imágenes, sea $\mathcal{D}$ un operador discretización (visto en la §2.2.2) entonces

$$\hat{\mathcal{P}}_{\mathcal{Z}} = \arg\min_{g\in\mathcal{K}} \sum_{f\in\mathcal{Z}} \sum_{k=1}^{N} \sum_{i,j\in\mathcal{M}_k} L((\mathcal{D}_k f)_{2i,2j}, g(\mathcal{S}^{r,s}(\mathcal{D}_{k-1} f)_{i,j})).$$

Es decir tomamos un conjunto de señales o imágenes, "unimos" todos los todos los niveles y todos los posibles valores, los utilizamos como conjunto de entrenamiento y obtenemos un filtro general.

[3]Del mismo modo que hemos hecho en la §4.3.1 podemos combinar ambos métodos obteniendo filtros para cada orientación con independencia del nivel de multiresolución, así quedaría:

$$(\hat{\mathcal{P}}_{\rightarrow f}) = \arg\min_{g\in\mathcal{K}} \sum_{k=1}^{N} \sum_{i,j\in\Gamma_{\rightarrow}^{k-1}} L(f_{2i,2j}^{k}, g(\mathcal{S}^{r,s}(f_{i,j}^{k-1}))),$$

$$(\hat{\mathcal{P}}_{\downarrow f}) = \arg\min_{g\in\mathcal{K}} \sum_{k=1}^{N} \sum_{i,j\in\Gamma_{\downarrow}^{k-1}} L(f_{2i,2j}^{k}, g(\mathcal{S}^{r,s}(f_{i,j}^{k-1}))),$$

$$(\hat{\mathcal{P}}_{\searrow f}) = \arg\min_{g\in\mathcal{K}} \sum_{k=1}^{N} \sum_{i,j\in\Gamma_{\searrow}^{k-1}} L(f_{2i,2j}^{k}, g(\mathcal{S}^{r,s}(f_{i,j}^{k-1}))),$$

$$(\hat{\mathcal{P}}_{\swarrow f}) = \arg\min_{g\in\mathcal{K}} \sum_{k=1}^{N} \sum_{i,j\in\Gamma_{\swarrow}^{k-1}} L(f_{2i,2j}^{k}, g(\mathcal{S}^{r,s}(f_{i,j}^{k-1}))),$$

$$\hat{\mathcal{P}}_f = \arg\min_{g\in\mathcal{K}} \sum_{k=1}^{N} \sum_{i,j\in\Gamma^{k-1}} L(f_{2i,2j}^{k}, g(\mathcal{S}^{r,s}(f_{i,j}^{k-1}))).$$

A este método le llamaremos *Learning-based multiresolution edge and non level dependent*:LM-nld-ed.

Del mismo modo podríamos pensar *¿es posible encontrar el mejor filtro para cualquier señal o para cualquier imagen (con dependencia de la función de pérdida, $L(x,y)$, y del conjunto de funciones donde minimizar, $\mathcal{K}$)?*

El problema es que si utilizamos un conjunto de imágenes muy amplio para construir un operador predicción gran parte de los elementos del conjunto de entrenamiento serán elementos pertenecientes a la parte continua de una imagen, es decir a aquella parte que no es un contorno. Esto provocará que nuestro filtro cada vez se aproxime más al filtro basado en interpolación segmentaria. Incluso obtendremos peores resultados pues tenemos que cuantizar este filtro para poder almacenarlo.

4.3.3 Tercer problema: consistencia en el esquema de multiresolución

El problema de la consistencia del esquema de multiresolución lo tratamos con detalle en la §2.3: En valores puntuales no tendremos problema ya que definimos el operador predicción en los valores pares como $(\mathcal{P}_{k-1}^k f^{k-1})_{2i,2j} = f_{i,j}^{k-1}$ y tenemos asegurada la consistencia. Sin embargo en medias en celda no tenemos necesariamente consistencia en los operadores. En ese caso utilizaremos la estrategia (AY).

4.3.4 Ejemplo de multiresolución de aprendizaje

Vamos a hallar los filtros correspondientes a una imagen. El operador decimación será él definido en el contexto de medias en celda. Las variables de nuestro problema son:

- Imagen f^N, *lena*: imagen en escala de grises, rango $[0, 255]$ de tamaño 512×512, así $N = 9$. Hallaremos los operadores predicción desde el nivel 2^5.

- Utilizaremos la norma $L(x,y) = |x-y|^2$. El mismo razonamiento que hacemos con norma 2 podríamos hacerlo con norma p, Huber, con $L_{dea,\kappa}$ o $L_{log,\kappa}$.

- Tomaremos una ventana de píxeles de tamaño 3×3, por tanto para

$2 \leq i, j \leq 2^k - 1$ tenemos que:

$$\mathcal{S}^{1,1}(f_{i,j}^{k-1}) = (f_{i-1,j-1}^{k-1}, f_{i-1,j}^{k-1}, f_{i-1,j+1}^{k-1}, f_{i,j-1}^{k-1}, f_{i,j}^{k-1}, f_{i,j+1}^{k-1}, f_{i+1,j-1}^{k-1}, f_{i+1,j}^{k-1}, f_{i+1,j+1}^{k-1}).$$

- Sea la clase de funciones $\mathcal{K} = \Pi_9^1(\mathbb{R})^*$.

Así pues siguiendo el Ejemplo 4.3 para resolver nuestro problema LM (*Learning-based multiresolution level-dependent*) tendríamos que minimizar los siguientes funcionales:

$$(\hat{\mathcal{P}}_{0,0})_{k-1}^k = \arg\min_{g \in \mathcal{K}} \sum_{i,j=2}^{J_{k-1}-1} L(f_{2i,2j}^k, g(\mathcal{S}^{1,1}(f_{i,j}^{k-1}))),$$

$$(\hat{\beta}_{0,0})_{k-1}^k = \min_{\beta \in \mathbb{R}^9} \sum_{i,j=2}^{J_{k-1}-1} |f_{2i,2j}^k - \mathcal{S}^{1,1}(f_{i,j}^{k-1})\beta^T|^2,$$

si denotamos

$$\mathbf{F}_{0,0}^{k-1} = \begin{pmatrix} f_{1,1}^{k-1} & f_{1,2}^{k-1} & \cdots & f_{1,J_{k-1}-2}^{k-1} & f_{2,1}^{k-1} & \cdots & f_{J_{k-1}-2,1}^{k-1} & \cdots & f_{J_{k-1}-2,J_{k-1}-2}^{k-1} \\ f_{1,2}^{k-1} & f_{1,3}^{k-1} & \cdots & f_{1,J_{k-1}-1}^{k-1} & f_{2,2}^{k-1} & \cdots & f_{J_{k-1}-2,2}^{k-1} & \cdots & f_{J_{k-1}-2,J_{k-1}-1}^{k-1} \\ f_{1,3}^{k-1} & f_{2,1}^{k-1} & \cdots & f_{1,J_{k-1}}^{k-1} & f_{2,3}^{k-1} & \cdots & f_{J_{k-1}-2,3}^{k-1} & \cdots & f_{J_{k-1}-2,J_{k-1}}^{k-1} \\ f_{2,1}^{k-1} & f_{2,2}^{k-1} & \cdots & f_{2,J_{k-1}-2}^{k-1} & f_{3,1}^{k-1} & \cdots & f_{J_{k-1}-1,1}^{k-1} & \cdots & f_{J_{k-1}-1,J_{k-1}-2}^{k-1} \\ f_{2,2}^{k-1} & f_{2,3}^{k-1} & \cdots & f_{2,J_{k-1}-1}^{k-1} & f_{3,2}^{k-1} & \cdots & f_{J_{k-1}-1,2}^{k-1} & \cdots & f_{J_{k-1}-1,J_{k-1}-1}^{k-1} \\ f_{2,3}^{k-1} & f_{3,1}^{k-1} & \cdots & f_{2,J_{k-1}}^{k-1} & f_{3,3}^{k-1} & \cdots & f_{J_{k-1}-1,3}^{k-1} & \cdots & f_{J_{k-1}-1,J_{k-1}}^{k-1} \\ f_{3,1}^{k-1} & f_{3,2}^{k-1} & \cdots & f_{3,J_{k-1}-2}^{k-1} & f_{4,1}^{k-1} & \cdots & f_{J_{k-1},1}^{k-1} & \cdots & f_{J_{k-1},J_{k-1}-2}^{k-1} \\ f_{3,2}^{k-1} & f_{3,3}^{k-1} & \cdots & f_{3,J_{k-1}-1}^{k-1} & f_{4,2}^{k-1} & \cdots & f_{J_{k-1},2}^{k-1} & \cdots & f_{J_{k-1},J_{k-1}-1}^{k-1} \\ f_{3,3}^{k-1} & f_{4,1}^{k-1} & \cdots & f_{3,J_{k-1}}^{k-1} & f_{4,3}^{k-1} & \cdots & f_{J_{k-1},3}^{k-1} & \cdots & f_{J_{k-1},J_{k-1}}^{k-1} \end{pmatrix}^T;$$

con

$$F_{0,0}^k = (f_{2,2}^k, \ldots, f_{2,J_k-2}^k, \ldots, f_{J_k-2,2}^k, \ldots, f_{J_k-2,J_k-2}^k)^T,$$

tenemos que

$$(\hat{\beta}_{0,0})_{k-1}^k = \min_{\beta \in \mathbb{R}^9} ||F_{0,0}^k - \mathbf{F}_{0,0}^{k-1}\beta^T||_2.$$

De forma análoga definiríamos cada uno de los *filtros* para los píxeles $(2i-1, 2j)$, $(2i, 2j-1)$ y calcularíamos el último valor por consistencia utilizando la estrategia (E0) vista en la §2.3:

$$(\hat{\beta}_{-1,0})_{k-1}^k = \min_{\beta \in \mathbb{R}^9} ||F_{-1,0}^k - \mathbf{F}_{-1,0}^{k-1}\beta^T||_2,$$

$$(\hat{\beta}_{-1,0})_{k-1}^k = \min_{\beta \in \mathbb{R}^9} ||F_{0,-1}^k - \mathbf{F}_{0,-1}^{k-1}\beta^T||_2,$$

$$(\hat{\beta}_{-1,-1})_{k-1}^k = 4(\delta_{m,5})_{m=1,\ldots,9} - ((\hat{\beta}_{0,0})_{k-1}^k + (\hat{\beta}_{-1,0})_{k-1}^k + (\hat{\beta}_{0,-1})_{k-1}^k).$$

Sin embargo si nosotros deseamos hallar un único *filtro* para todos los niveles de multiresolución, i.e. $\forall\, k$, (*Learning-based multiresolution non-level-dependent* §4.3.2) tendríamos:

$$\begin{aligned}
\hat{\beta}_{0,0} &= \min_{\beta\in\mathbb{R}^9} ||F_{0,0} - \mathbf{F}_{0,0}\beta^T||_2,\\
\hat{\beta}_{-1,0} &= \min_{\beta\in\mathbb{R}^9} ||F_{-1,0} - \mathbf{F}_{-1,0}\beta^T||_2,\\
\hat{\beta}_{-1,0} &= \min_{\beta\in\mathbb{R}^9} ||F_{0,-1} - \mathbf{F}_{0,-1}\beta^T||_2,\\
\hat{\beta}_{-1,-1} &= 4(\delta_{m,5})_{m=1,\ldots,9} - (\hat{\beta}_{0,0} + \hat{\beta}_{-1,0} + \hat{\beta}_{0,-1}),
\end{aligned}$$

donde $F_{0,0} = (F^6_{0,0}; F^7_{0,0}; F^8_{0,0}; F^9_{0,0})^T$ y $\mathbf{F}_{0,0} = (\mathbf{F}^5_{0,0}; \mathbf{F}^6_{0,0}; \mathbf{F}^7_{0,0}; \mathbf{F}^8_{0,0})^T$. Análogo para $F_{-1,0}, \mathbf{F}_{-1,0}$ y $F_{0,-1}, \mathbf{F}_{0,-1}$.

Veamos los *filtros* obtenidos donde $a(x) = [256x]$ siendo $[\cdot]$ el valor entero.

	$(\hat{\beta}_{0,0})^6_5$	$(\hat{\beta}_{-1,0})^6_5$	$(\hat{\beta}_{0,-1})^6_5$	$(\hat{\beta}_{-1,-1})^6_5$
$i-1, j-1$	0,04608280	−0,00041166	0,01495164	−0,06062278
$i-1, j$	−0,13450536	0,08438377	−0,16692559	0,21704718
$i-1, j+1$	−0,07514155	0,09664665	0,04161335	−0,06311845
$i, j-1$	−0,16204699	−0,19396066	0,07432917	0,28167848
i, j	0,92998846	1,12350047	1,04421949	0,90229158
$i, j+1$	0,27745737	0,05073476	−0,21474568	−0,11344645
$i+1, j-1$	−0,01473615	0,05827300	0,06468704	−0,10822389
$i+1, j$	0,18984069	−0,24056706	0,11760001	−0,06687365
$i+1, j+1$	−0,05785589	0,02143769	0,02473592	0,01168227
	$a((\hat{\beta}_{0,0})^6_5)$	$a((\hat{\beta}_{-1,0})^6_5)$	$a((\hat{\beta}_{0,-1})^6_5)$	$a((\hat{\beta}_{-1,-1})^6_5)$
$i-1, j-1$	12	−0	4	−16
$i-1, j$	−34	22	−43	56
$i-1, j+1$	−19	25	11	−16
$i, j-1$	−41	−50	19	72
i, j	238	288	267	231
$i, j+1$	71	13	−55	−29
$i+1, j-1$	−4	15	17	−28
$i+1, j$	49	−62	30	−17
$i+1, j+1$	−15	5	6	3

	$(\hat{\beta}_{0,0})_6^7$	$(\hat{\beta}_{-1,0})_6^7$	$(\hat{\beta}_{0,-1})_6^7$	$(\hat{\beta}_{-1,-1})_6^7$
$i-1, j-1$	0,01580854	0,00583569	0,02478468	−0,04642890
$i-1, j$	−0,11556991	0,10417350	−0,21462816	0,22602458
$i-1, j+1$	−0,05946923	0,07325312	0,04864424	−0,06242813
$i, j-1$	−0,11015622	−0,22197263	0,09575875	0,23637009
i, j	0,92079097	1,08162369	1,07668620	0,92089915
$i, j+1$	0,25557590	0,06933151	−0,19676384	−0,12814357
$i+1, j-1$	−0,07225361	0,06325316	0,04999575	−0,04099530
$i+1, j$	0,21506517	−0,19957810	0,12251280	−0,13799986
$i+1, j+1$	−0,05050948	0,02470480	−0,00690517	0,03270985
	$a((\hat{\beta}_{0,0})_6^7)$	$a((\hat{\beta}_{-1,0})_6^7)$	$a((\hat{\beta}_{0,-1})_6^7)$	$a(\widehat{(\beta_{-1,-1})_6^7})$
$i-1, j-1$	4	1	6	−12
$i-1, j$	−30	27	−55	58
$i-1, j+1$	−15	19	12	−16
$i, j-1$	−28	−57	25	61
i, j	236	277	276	236
$i, j+1$	65	18	−50	−33
$i+1, j-1$	−18	16	13	−10
$i+1, j$	55	−51	31	−35
$i+1, j+1$	−13	6	−2	8

	$(\hat{\beta}_{0,0})_7^8$	$(\hat{\beta}_{-1,0})_7^8$	$(\hat{\beta}_{0,-1})_7^8$	$(\hat{\beta}_{-1,-1})_7^8$
$i-1, j-1$	0,05551809	−0,01009141	0,01072150	−0,05614819
$i-1, j$	−0,15986791	0,10044151	−0,18192289	0,24134928
$i-1, j+1$	−0,04579442	0,07218444	0,03636262	−0,06275265
$i, j-1$	−0,14703276	−0,16521786	0,07718401	0,23506661
i, j	0,94509975	1,04449186	1,07782353	0,93258485
$i, j+1$	0,24936174	0,10089050	−0,18981687	−0,16043538
$i+1, j-1$	−0,05129705	0,03623127	0,06297475	−0,04790897
$i+1, j$	0,20112099	−0,18343620	0,11830338	−0,13598817
$i+1, j+1$	−0,04695374	0,00441437	−0,01171627	0,05425564
	$a((\hat{\beta}_{0,0})_7^8)$	$a((\hat{\beta}_{-1,0})_7^8)$	$a((\hat{\beta}_{0,-1})_7^8)$	$a((\hat{\beta}_{-1,-1})_7^8)$
$i-1, j-1$	14	−3	3	−14
$i-1, j$	−41	26	−47	62
$i-1, j+1$	−12	18	9	−16
$i, j-1$	−38	−42	20	60
i, j	242	267	276	239
$i, j+1$	64	26	−49	−41
$i+1, j-1$	−13	9	16	−12
$i+1, j$	51	−47	30	−35
$i+1, j+1$	−12	1	−3	14

	$(\hat{\beta}_{0,0})_8^9$	$(\hat{\beta}_{-1,0})_8^9$	$(\hat{\beta}_{0,-1})_8^9$	$(\hat{\beta}_{-1,-1})_8^9$
$i-1, j-1$	0,06290103	−0,02579248	−0,00788344	−0,02922511
$i-1, j$	−0,14689046	0,15212430	−0,19235748	0,18712363
$i-1, j+1$	−0,05945672	0,03128771	0,05732403	−0,02915503
$i, j-1$	−0,18852058	−0,18167336	0,13687821	0,23331573
i, j	0,96875364	1,02900586	1,03424604	0,96799446
$i, j+1$	0,23297722	0,13956696	−0,18654696	−0,18599722
$i+1, j-1$	−0,03056206	0,05355711	0,03375332	−0,05674837
$i+1, j$	0,19426012	−0,18255630	0,14188562	−0,15358944
$i+1, j+1$	−0,03337958	−0,01556170	−0,01727743	0,06621871
	$a((\hat{\beta}_{0,0})_8^9)$	$a((\hat{\beta}_{-1,0})_8^9)$	$a((\hat{\beta}_{0,-1})_8^9)$	$a((\hat{\beta}_{-1,-1})_8^9)$
$i-1, j-1$	16	−7	−2	−7
$i-1, j$	−38	39	−49	48
$i-1, j+1$	−15	8	15	−7
$i, j-1$	−48	−47	35	60
i, j	248	263	265	248
$i, j+1$	60	36	−48	−48
$i+1, j-1$	−8	14	9	−15
$i+1, j$	50	−47	36	−39
$i+1, j+1$	−9	−4	−4	17

El filtro general para todos los niveles, LM-nld es

	$\hat{\beta}_{0,0}$	$\hat{\beta}_{-1,0}$	$\hat{\beta}_{0,-1}$	$\hat{\beta}_{-1,-1}$
$i-1, j-1$	0,05512814	−0,01685047	0,00168490	−0,03996257
$i-1, j$	−0,14583513	0,12960189	−0,19181624	0,20804948
$i-1, j+1$	−0,05706711	0,04972981	0,05088224	−0,04354494
$i, j-1$	−0,16810916	−0,18309579	0,11377040	0,23743455
i, j	0,95487968	1,04410922	1,05178668	0,94922443
$i, j+1$	0,24168779	0,11758400	−0,19014623	−0,16912556
$i+1, j-1$	−0,03977849	0,05122305	0,04491685	−0,05636140
$i+1, j$	0,19854177	−0,18849861	0,13194926	−0,14199242
$i+1, j+1$	−0,03939753	−0,00381551	−0,01301746	0,05623050
	$a(\hat{\beta}_{0,0})$	$a(\hat{\beta}_{-1,0})$	$a(\hat{\beta}_{0,-1})$	$a(\hat{\beta}_{-1,-1})$
$i-1, j-1$	14	−4	0	−10
$i-1, j$	−37	33	−49	53
$i-1, j+1$	−15	13	13	−11
$i, j-1$	−43	−47	29	61
i, j	244	267	269	243
$i, j+1$	62	30	−49	−43
$i+1, j-1$	−10	13	11	−14
$i+1, j$	51	−48	34	−36
$i+1, j+1$	−10	−1	−3	14

El filtro general utilizando **23 imágenes** para todos los niveles es

	$\hat{\beta}_{0,0}$	$\hat{\beta}_{-1,0}$	$\hat{\beta}_{0,-1}$	$\hat{\beta}_{-1,-1}$
$i-1, j-1$	0,06867838	−0,0336359744	0,0003070275	−0,0353498297
$i-1, j$	0,08713471	0,1040019553	−0,1034303199	−0,0876870259
$i-1, j+1$	−0,02001792	0,0616797226	−0,0307891233	−0,0108706530
$i, j-1$	0,08304460	−0,0836174983	0,0912567739	−0,0906856659
i, j	1,00320915	1,0008691087	0,9942407940	1,0016523225
$i, j+1$	−0,08578106	0,0859903111	−0,0833901810	0,0831951116
$i+1, j-1$	−0,00588711	−0,0309779005	0,0602953117	−0,0234444097
$i+1, j$	−0,09102242	−0,1039035448	0,1021168155	0,0928227407
$i+1, j+1$	−0,03932722	−0,0006445803	−0,0304265050	0,0703925023
	$a(\hat{\beta}_{0,0})$	$a(\hat{\beta}_{-1,0})$	$a(\hat{\beta}_{0,-1})$	$a(\hat{\beta}_{-1,-1})$
$i-1, j-1$	18	−9	0	−9
$i-1, j$	22	27	−26	−22
$i-1, j+1$	−5	16	−8	−3
$i, j-1$	21	−21	23	−23
i, j	257	256	255	256
$i, j+1$	−22	22	−21	21
$i+1, j-1$	−2	−8	15	−6
$i+1, j$	−23	−27	26	24
$i+1, j+1$	−10	0	−8	18

4.3.5 Experimentos numéricos

En esta sección utilizaremos los algoritmos de multiresolución para la compresión de imágenes. Trabajaremos con dos imágenes reales: *lena* y *peppers* y una geométrica: *geopa*.

Como el resto de capítulos los experimentos dos dimensionales los haremos en el contexto de medias en celda (ver §2.2.2). La norma que utilizaremos será E_2 vista en los capítulos 2 y 3. Con esta norma definimos el valor PSNR también utilizado en los capítulos anteriores.

Cuanto mayor sea el valor PSNR menor será la distancia entre la imagen original y la imagen reconstruida. En muchas ocasiones a mayores PSNR no necesariamente tenemos mayor calidad visual, por esto mostraremos una comparación entre las diferentes imágenes obtenidas utilizando cada uno de los métodos.

Realizamos una comparación entre los métodos interpolatorios utilizando una ventana de 3×3 puntos, es decir, el producto tensorial del operador:

$$\begin{cases} (\mathcal{P}_{k-1}^{k} f^{k-1})_{2j-1} = \frac{1}{8} f_{j-1}^{k-1} + f_j^{k-1} - \frac{1}{8} f_{j+1}^{k-1}, \\ (\mathcal{P}_{k-1}^{k} f^{k-1})_{2j} = -\frac{1}{8} f_{j-1}^{k-1} + f_j^{k-1} + \frac{1}{8} f_{j+1}^{k-1}, \end{cases}$$

que denotaremos CA_3; una ventana 5×5 puntos, es decir el producto tensorial del operador:

$$\begin{cases} (\mathcal{P}_{k-1}^k f^{k-1})_{2j-1} = -\frac{3}{128} f_{j-2}^{k-1} + \frac{22}{128} f_{j-1}^{k-1} + f_j^{k-1} - \frac{22}{128} f_{j+1}^{k-1} + \frac{3}{128} f_{j+2}^{k-1}, \\ (\mathcal{P}_{k-1}^k f^{k-1})_{2j} = \frac{3}{128} f_{j-2}^{k-1} - \frac{22}{128} f_{j-1}^{k-1} + f_j^{k-1} + \frac{22}{128} f_{j+1}^{k-1} - \frac{3}{128} f_{j+1}^{k-1}, \end{cases}$$

que denotaremos como CA_5 (vistos en el capítulo 2) y los métodos LM con todas las posibilidades que describimos en este capítulo, es decir, LM dependiente de la escala que denotaremos como ^{p}LM; no dependiente del nivel de resolución ^{p}LM-nld (*non-level dependent*); LM *edge-dependent non-level dependent* que denotaremos como ^{p}LM-nld-ed y LM dependiente del contorno y del nivel ^{p}LM-ed siempre utilizando una ventana centrada de 3×3 puntos.

Para comprimir los detalles utilizamos la siguiente cuantización:

$$(\hat{d}^k)_{i,j} = 2\varepsilon_k \left[\frac{d_{i,j}^k}{2\varepsilon_k} \right], \tag{4.14}$$

donde d^k es el conjunto de detalles obtenidos en el paso k, con

$$\varepsilon_{k-1} = \begin{cases} \frac{\varepsilon_k}{2}, & \text{si} \quad \varepsilon_k \geq 1/2; \\ 1/2, & \text{si} \quad \varepsilon_k < 1/2 \end{cases} \tag{4.15}$$

siendo ε_N dado y $[\cdot]$ es el entero obtenido por redondeo.

Para medir la capacidad de compresión del método contabilizamos el número de elementos no nulos (denotado por NNZ) siendo cuanto menor número mayor compresión. En el caso de los métodos LM tendríamos que añadir a NNZ el número de elementos que guardamos NF como filtros. El número de filtros será $\text{NF} = 36L\text{NE}$, donde L es el número de predictores dependiendo del nivel, en el caso de LM-nld y LM-nld-ed $L = 1$, para LM y LM-ed $L = 4$ y NE es el número de orientaciones que tomamos (ver §4.3.2), en el caso de LM-nld y LM $\text{NE} = 1$ y en el caso de LM-nld-ed y LM-ed $\text{NE} = 5$. Mostramos en la Tabla 4.6 el número de elementos necesarios para cada método.

Cuando introducimos en los métodos LM la detección de contornos (LM-nld-ed y LM-ed) perdemos el control del error del método. Así pues tenemos que utilizar una estrategia propuesta por Aràndiga et al. en [2] que consiste en modificar el algoritmo e ir calculando en cada paso los detalles y cuantizándolos (ver capítulo 3 o ver para más detalles [3, 7]). Utilizaremos en todos los métodos esta estrategia para establecer una comparativa.

Método	NF
LM-nld	36
LM	144
LM-nld-ed	180
LM-ed	720

Tabla 4.6: *Número de filtros necesarios para los métodos LM.*

Los métodos LM mejoran la calidad de la imagen y disminuyen el número de elementos no nulos (NNZ) (Tablas 4.7, 4.8 y 4.9), sin embargo tenemos que almacenar los filtros. Los métodos CA y LM-ed tienen el mismo número de elementos no nulos (NNZ) y sin embargo LM-ed produce un mejor PSNR y una mejor calidad visual. Esto quiere decir que los elementos almacenados **contienen información más relevante**, los filtros están adaptados perfectamente a los contornos de las imágenes conteniendo en ellos información relevante sobre éstas.

No obtenemos gran diferencia entre utilizar los métodos para imágenes geométricas y para imágenes reales como podemos ver en las Figuras 4.14, 4.16 y 4.18.

En las Figuras 4.15, 4.17 y 4.19 ampliamos una zona de la imagen correspondiente. Se aprecia perfectamente que guardando menor cantidad de detalles obtenemos con LM-ed una mejor calidad visual.

En definitiva los métodos LM adaptados a contornos producen muy buenos resultados. El único problema que presentan es que tenemos que almacenar los filtros.

	$\varepsilon_N = 8$		$\varepsilon_N = 16$		$\varepsilon_N = 32$		$\varepsilon_N = 64$	
	NNZ	PSNR	NNZ	PSNR	NNZ	PSNR	NNZ	PSNR
CA_3	19029	34,08	8926	30,93	4237	27,87	1598	24,74
CA_5	18840	34,03	8767	30,84	4094	27,80	1565	24,65
^{2}LM-nld	19637	33,83	9059	30,64	4230	27,65	1601	24,53
^{2}LM	19753	34,11	9115	31,06	4315	28,20	1620	25,18
^{2}LM-nld-ed	18900	33,95	8738	30,83	4081	27,83	1557	24,66
^{2}LM-ed	18987	34,32	8863	31,42	4104	28,68	1553	25,65
^{1}LM-nld	18973	34,03	8900	30,87	4189	27,81	1594	24,69
^{1}LM	19165	34,21	8938	31,14	4183	28,23	1586	25,18
^{1}LM-nld-ed	18540	33,77	8589	30,74	4072	27,82	1560	24,68
^{1}LM-ed	18548	34,33	8604	31,43	4012	28,63	1526	25,58

Tabla 4.7: *Imagen* peppers. *Resultados utilizando multiresolución con métodos LM con* $\varepsilon_N = 8, 16, 32, 64$.

	$\varepsilon_N = 8$		$\varepsilon_N = 16$		$\varepsilon_N = 32$		$\varepsilon_N = 64$	
	NNZ	PSNR	NNZ	PSNR	NNZ	PSNR	NNZ	PSNR
CA_3	6337	42,03	2876	36,73	1541	33,39	589	29,48
CA_5	6218	41,43	2836	36,64	1444	33,15	572	29,23
^{2}LM-nld	6541	41,01	3727	36,29	1495	32,73	632	28,82
^{2}LM	6868	41,37	3516	36,33	1679	33,41	580	29,68
^{2}LM-nld-ed	5389	42,31	3324	37,13	1434	33,17	575	29,26
^{2}LM-ed	5433	43,19	2961	38,73	1435	34,98	405	31,13
^{1}LM-nld	4958	41,94	2990	36,85	1525	33,33	756	29,65
^{1}LM	5905	41,60	2949	37,35	1498	33,43	503	29,50
^{1}LM-nld-ed	4482	43,00	2847	38,21	1472	33,65	700	29,03
^{1}LM-ed	4253	42,86	2727	39,11	1271	34,90	431	31,22

Tabla 4.8: *Imagen* geopa. *Resultados utilizando multiresolución con métodos LM con* $\varepsilon_N = 8, 16, 32, 64$.

	$\varepsilon_N = 8$		$\varepsilon_N = 16$		$\varepsilon_N = 32$		$\varepsilon_N = 64$	
	NNZ	PSNR	NNZ	PSNR	NNZ	PSNR	NNZ	PSNR
CA_3	16481	34,36	7202	31,28	2772	28,32	908	25,79
CA_5	15758	34,38	6957	31,31	2676	28,31	853	25,72
^{2}LM-nld	16278	34,23	7044	31,11	2718	28,20	846	25,64
^{2}LM	16347	34,36	7097	31,44	2727	28,74	849	26,22
^{2}LM-nld-ed	15757	34,25	6902	31,19	2626	28,24	823	25,71
^{2}LM-ed	15888	34,54	6876	31,74	2631	29,06	794	26,52
^{1}LM-nld	16097	34,26	7011	31,17	2711	28,26	847	25,68
^{1}LM	16153	34,42	7019	31,48	2698	28,73	857	26,23
^{1}LM-nld-ed	15661	34,20	6849	31,17	2619	28,24	826	25,71
^{1}LM-ed	15665	34,58	6820	31,72	2576	28,96	805	26,53

Tabla 4.9: *Imagen* lena*. Resultados utilizando multiresolución con métodos LM con* $\varepsilon_N = 8, 16, 32, 64$*.*

4.4 Conclusiones y futuras líneas de investigación

Presentamos en este capítulo (ver también p. ej. [11, 10]) una nueva manera de hallar el operador predicción. Por primera vez se toma en cuenta aquellos valores que se van a predecir introduciendo esta información en el operador. Es decir, se traspasa información de los errores al operador predicción. Evidentemente debemos asegurar que este traspaso nos permite una mayor compresión de datos, menor número de detalles. Se introduce un problema nuevo con varias componentes, cada una de estas componentes proporcionará un nuevo operador con distintas propiedades:

- La función de pérdida $L(x, y)$. La precisión del operador dependerá de esta elección.
- La clase de funciones que tomemos determina el orden, el control del error y la compresión.

En la práctica no hemos encontrado una clase de funciones no lineal donde obtengamos un operador predicción que produzca mejores resultados que el operador basado en interpolación polinómica.

CA_3 PSNR: 27,87, NNZ: 4237

CA_5, PSNR: 27,80, NNZ: 4094

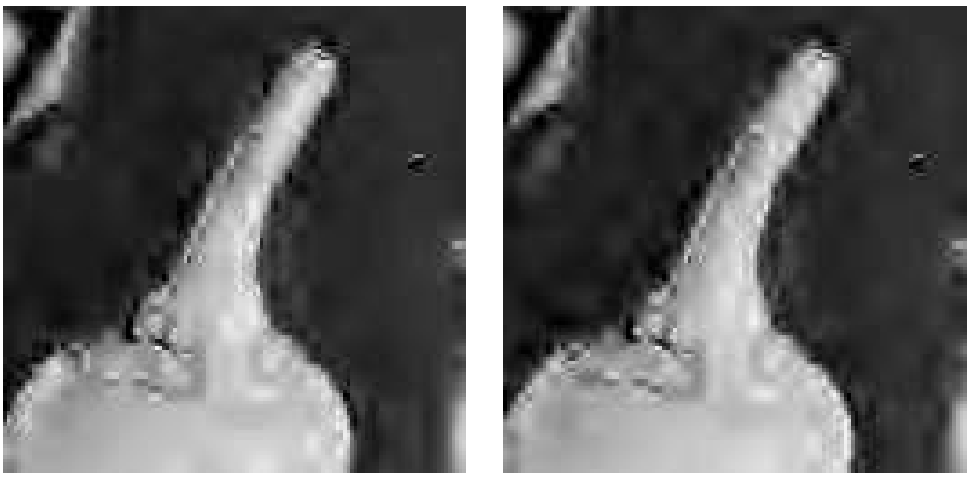
^{2}LM PSNR: 28,20, NNZ: 4315

^{1}LM, PSNR: 28,23, NNZ: 4183

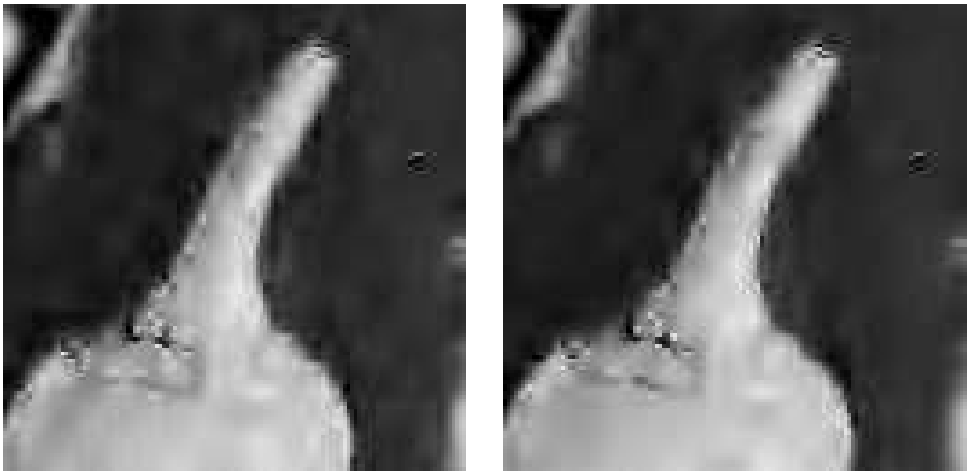
^{2}LM-ed, PSNR: 28,68, NNZ: 4104

^{1}LM-ed, PSNR: 28,63, NNZ: 4012

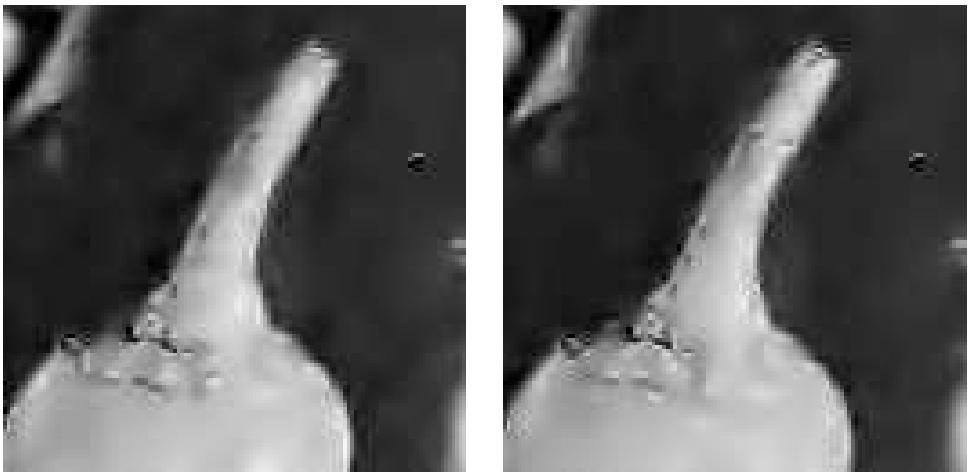
Figura 4.14: *Imágenes resultantes después de aplicar el algoritmo de compresión con la técnica (AY) y los operadores predictor* ^{1,2}LM, ^{1,2}LM-ed *para* peppers.



CA_3 PSNR: 27,87, NNZ: 4237 CA_5, PSNR: 27,80, NNZ: 4094



^{2}LM PSNR: 28,20, NNZ: 4315 ^{1}LM, PSNR: 28,23, NNZ: 4183

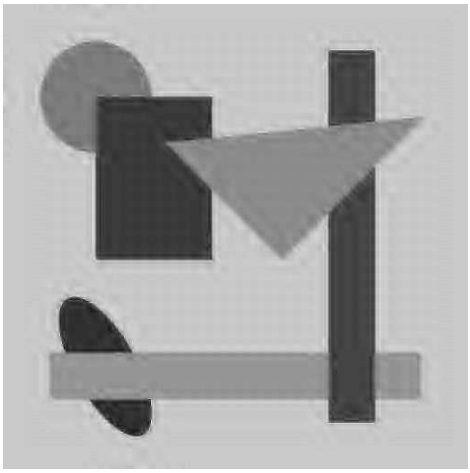


^{2}LM-ed, PSNR: 28,68, NNZ: 4104 ^{1}LM-ed, PSNR: 28,63, NNZ: 4012

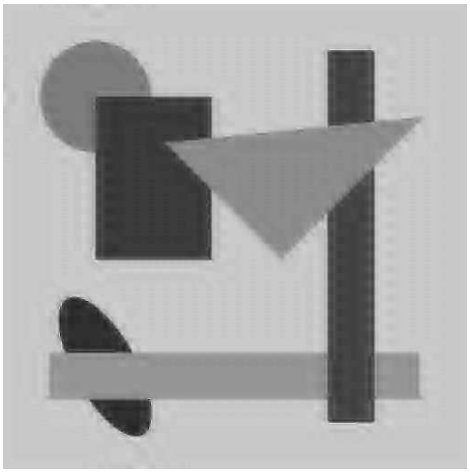
Figura 4.15: *Zoom de las imágenes resultantes después de aplicar el algoritmo de compresión con la técnica (AY) y los operadores predictor* ^{1,2}LM, ^{1,2}LM-ed *para* peppers.

CA_3, PSNR: 33,39, NNZ: 1541

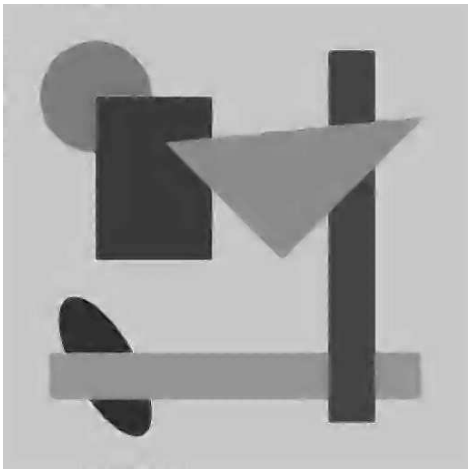
CA_5, PSNR: 33,15, NNZ: 1444



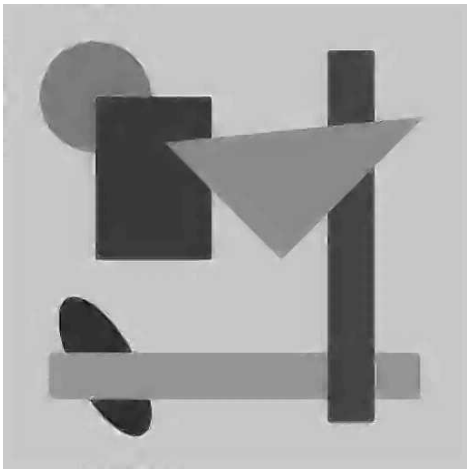
^{2}LM, PSNR: 33,41, NNZ: 1679



^{1}LM, PSNR: 33,43, NNZ: 1498

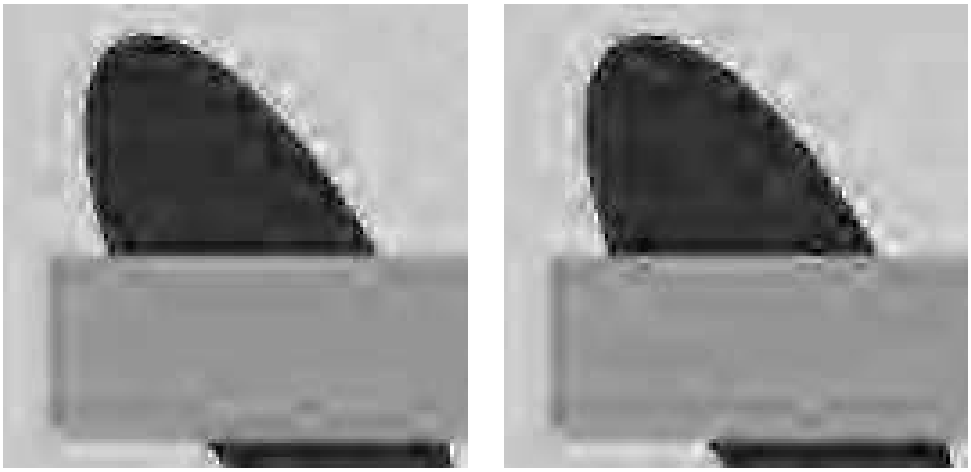


^{2}LM-ed, PSNR: 34,98, NNZ: 1435

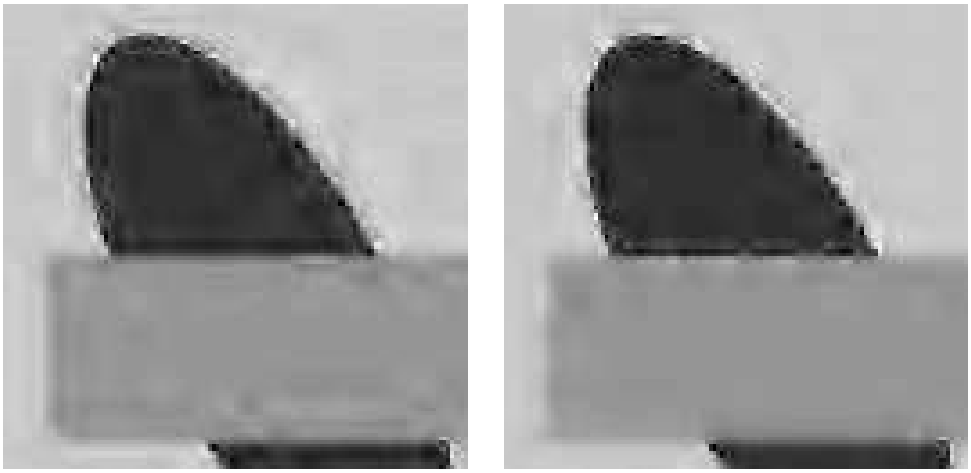


^{1}LM-ed, PSNR: 34,90, NNZ: 1271

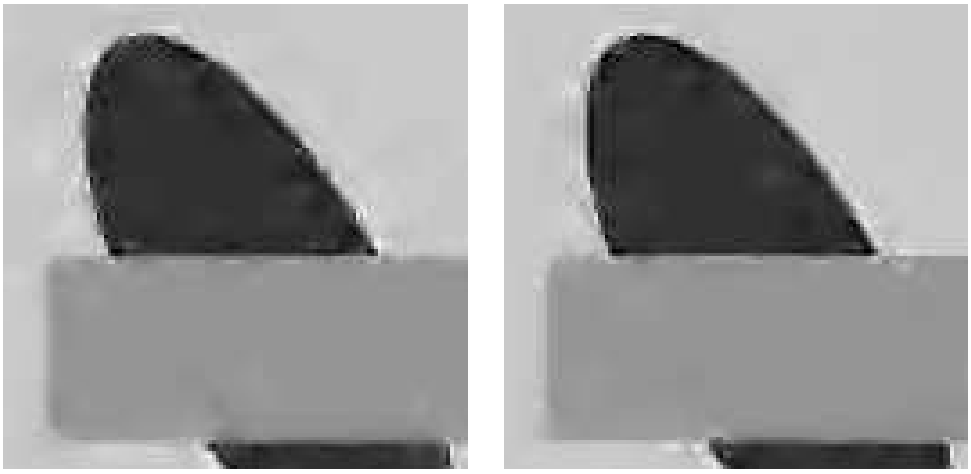
Figura 4.16: *Imágenes resultantes después de aplicar el algoritmo de compresión con la técnica (AY) y los operadores predictor* ^{1,2}LM, ^{1,2}LM-ed *para* geopa.



CA_3, PSNR: 33,39, NNZ: 1541 CA_5, PSNR: 33,15, NNZ: 1444



^{2}LM, PSNR: 33,41, NNZ: 1679 ^{1}LM, PSNR: 33,43, NNZ: 1498



^{2}LM-ed, PSNR: 34,98, NNZ: 1435 ^{1}LM-ed, PSNR: 34,90, NNZ: 1271

Figura 4.17: *Zoom de las imágenes resultantes después de aplicar el algoritmo de compresión con la técnica (AY) y los operadores predictor* ^{1,2}LM, ^{1,2}LM-ed *para* geopa.

CA_3, PSNR: 28,32, NNZ: 2772

CA_5, PSNR: 28,31, NNZ: 2676

^{2}LM, PSNR: 28,74, NNZ: 2727

^{1}LM, PSNR: 28,73, NNZ: 2698

^{2}LM-ed, PSNR: 29,06, NNZ: 2631

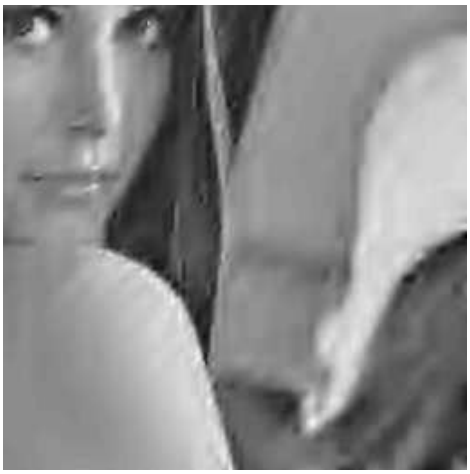
^{1}LM-ed, PSNR: 28,96, NNZ: 2576

Figura 4.18: *Imágenes resultantes después de aplicar el algoritmo de compresión con la técnica (AY) y los operadores predictor* ^{1,2}LM, ^{1,2}LM-ed *para* lena.

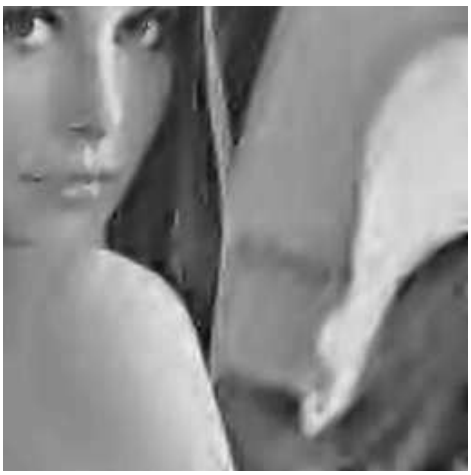
CA_3, PSNR: 28,32, NNZ: 2772

CA_5, PSNR: 28,31, NNZ: 2676

^{2}LM, PSNR: 28,74, NNZ: 2727



^{1}LM, PSNR: 28,73, NNZ: 2698



^{2}LM-ed, PSNR: 29,06, NNZ: 2631

^{1}LM-ed, PSNR: 28,96, NNZ: 2576

Figura 4.19: *Zoom de las imágenes resultantes después de aplicar el algoritmo de compresión con la técnica (AY) y los operadores predictor* ^{1,2}LM, ^{1,2}LM-ed *para* lena.

En los ejemplos uno dimensionales la elección de la función de pérdida es determinante. Hemos obtenido muy buenos resultados para funciones oscilantes con cualquier tipo de función de pérdida. Para funciones que presentan diferentes zonas: oscilantes, no oscilantes o discontinuidades hemos obtenido muy buenos resultados con la función robusta L_1 minimizándola utilizando la técnica presentada por Vogel en [91] y cuya convergencia fue estudiada por T. Chan et al. en [26].

Esto nos lleva a pensar como futura línea de investigación que podemos hacer algoritmos de compresión con zonas sin pérdida utilizando los métodos LM.

Otra característica a destacar es que al contrario que sucede con los métodos habituales en la construcción del operador predicción no influye cuál sea el operador discretización (o decimación) ya que el funcional se plantea del mismo modo sea cual sea la discretización.

Como aspectos negativos del método, solo lo podemos utilizar si conocemos todos los niveles de multiresolución al contrario que los métodos habituales (ver p. ej. [13]) o los métodos presentados en el capítulo 3. Por otra parte debemos almacenar el predictor lo que acarrea un coste en la compresión.

Los resultados obtenidos en compresión de imágenes son buenos (sobretodo visualmente) cuando introducimos la detección de los contornos.

El método abre las puertas a la utilización de otros métodos como redes neuronales (ver capítulo 11 de [56] o [19, 78]) o métodos de árbol.

Conclusiones generales y líneas futuras de investigación

El objetivo fundamental de este trabajo es conseguir comprimir señales o imágenes digitales utilizando multiresolución *à la Harten* que es una generalización de las bases *wavelets* (ver p.ej. [34, 72]). Como vimos en el capítulo 2 el marco general presentado por Harten se basa en dos operadores: un operador decimación $\mathcal{D}_k^{k-1}$ que es lineal y sobreyectivo y un operador predicción $\mathcal{P}_{k-1}^k$ que junto con el anterior satisfacen la condición de consistencia, es decir $\mathcal{D}_k^{k-1}\mathcal{P}_{k-1}^k = I_{V^{k-1}}$.

En este trabajo introducimos una nueva estrategia que permite utilizar la multiresolución diseñando operadores no consistentes bajo el contexto de medias en celda obteniendo buenos resultados a nivel teórico (conserva orden y tenemos control del error) y práctico. Una posible línea de investigación sería estudiar el comportamiento de esta estrategia para otros operadores decimación. Además nos va a permitir generalizar a medias en celda el método no lineal basado en medias PPH explicado en la §2.2.3 (también ver p. ej. [5, 4]) de una forma natural basándonos en esquemas de subdivisión B-*spline* (ver p. ej. [33]).

Esta estrategia surge como solución a la problemática que supone el hallar operadores predicción utilizando métodos de núcleo que explicamos en el capítulo 3. Este tipo de multiresolución generaliza los métodos clásicos utilizando interpolación polinómica segmentaria (ver p. ej. [15]). Presentamos estos métodos para valores puntuales y para medias en celda. Demostramos la linealidad del operador si utilizamos norma 2 en la función de pérdida. No podemos afirmar nada sobre la estabilidad

al utilizar otro tipo de normas (norma 1 u otras), así pues una posible línea de investigación sería estudiar las propiedades teóricas al utilizar normas diferentes a la norma 2.

Con estos métodos (junto con la estrategia diseñada) obtenemos buenos resultados a nivel práctico tanto en compresión de señales como de imágenes. Al generalizar a medias en celda definimos una pseudodistancia entre celdas que provoca ciertos resultados, otra posible vía de investigación sería ver las consecuencias de elegir otro tipo de distancias (o pseudodistancia) para conjuntos como la distancia de Hausdorff. Sería interesante ver los métodos de núcleo para otros operadores discretización. Hemos tomado en este trabajo para aproximar el conjunto de polinomios, podría utilizarse otra clase de funciones $\mathcal{K}$. La combinación de este método con los métodos no lineales como ENO y WENO puede ser también de gran utilidad.

Por otra parte, en la §3.5 bajo el contexto de métodos de núcleo podemos ver un tipo diferente de aproximación (llamada "suavizado a un lado") que marca las discontinuidades aisladas (ver [74, 47, 70]). Los resultados con este método son sorprendentes pero no relevantes pues utilizamos una discontinuidad aislada. Podríamos estudiar las posibilidades que nos ofrecen estos métodos en multiresolución.

El campo de los métodos de núcleo es muy amplio, nosotros lo hemos "*unido*" con las técnicas de multiresolución así pues queda abierto un mundo donde poder explorar las distintas alternativas.

A partir del capítulo 4 hemos dado un giro a la multiresolución utilizando los valores que vamos a predecir para hallar los predictores, es decir: **aprender a predecir**. Para ello hemos utilizado teoría estadística de aprendizaje. Hemos intentado calcular el mejor filtro para el conjunto de todas las imágenes y hemos obtenido que este filtro, si tomamos imágenes aleatorias, es simplemente la réplica del píxel. Sin embargo, si tomamos un conjunto de imágenes reales tiende al filtro obtenido por interpolación lineal (lo que marca la continuidad de las imágenes reales). Estos resultados no han sido de gran relevancia. No obstante si aprendemos a predecir a nivel local, es decir dependiendo tan sólo de la imagen, e incluso de los contornos de la imagen obtenemos resultados interesantes sobretodo en cuanto a calidad visual se refiere. Hemos probado el orden y el control del error para este método.

En esta memoria tan solo hemos utilizado la clase de funciones lineales. Sería interesante estudiar las posibilidades utilizando otro tipo de funciones. Hemos minimizado utilizando gran variedad de normas introduciendo programación lineal sin obtener grandes resultados exceptuando en norma 1. Otra posible vía de investigación es trabajar con

diferentes problemas de programación lineal, es decir, las condiciones que hemos puesto han sido simplemente que el predictor reduzca la norma p del error, esto provoca que cada componente del error sea pequeña de manera que la podemos cuantizar, sin embargo podemos establecer otras condiciones que sean adecuadas y adaptadas al problema que nos presenten.

El campo de la teoría de aprendizaje es muy extenso, se pueden utilizar como mencionábamos en el capítulo 4 métodos de árbol, redes neuronales y otras muchas técnicas. Avanzar en el estudio de estas técnicas podría ser interesante.

En definitiva hemos presentado en este trabajo dos nuevas formas de diseñar un operador predicción dentro de un esquema de multiresolución, éstos nos han producido diversos problemas que hemos ido solucionando y que nos han abierto puertas hacia nuevos problemas.

Conclusiones

Esquematizamos las conclusiones obtenidas:

a) Hemos generalizado las técnicas no lineales ENO y WENO para calcular la aproximación a las derivadas con el objetivo de realizar un operador de Hermite monótono.

b) Los resultados de utilizar ENO y WENO en la construcción de un operador monótono de Hermite han sido satisfactorios.

c) Hemos desarrollado una estrategia para poder utilizar operadores no consistentes en multiresolución que hemos llamado (AY) obteniendo buenos resultados a nivel teórico y práctico.

d) Hemos introducido los métodos de núcleo dentro de un esquema de multiresolución, generalizando los métodos de interpolación polinómica segmentaria obteniendo buenos resultados a nivel teórico y práctico, superando a los métodos lineales vistos hasta ahora.

e) Hemos generalizado a medias en celda la multiresolución basada en valores puntuales definiendo una pseudodistancia entre celdas.

f) Hemos sustituido la norma 2 en el funcional a minimizar por la norma p en los métodos de núcleo obteniendo buenos resultados en una dimensión, no lo hemos realizado para varias dimensiones pues es computacionalmente lento. No podemos afirmar nada sobre la estabilidad utilizando otras normas que no sean la norma 2.

g) Hemos generalizado a dos dimensiones los métodos de núcleo tanto para valores puntuales como para medias en celda obteniendo buenos resultados en la compresión de imágenes digitales.

h) Los resultados obtenidos para métodos de núcleo dependen del núcleo que tomemos y del peso que dé éste a las celdas próximas siendo mejores los núcleos concentrados en su centro y con una bajada pronunciada.

i) Hemos dado un giro a la multiresolución utilizando los valores a predecir para construir el predictor, para esto nos hemos basado en la teoría estadística de aprendizaje. A este tipo de multiresolución la hemos llamado LM.

j) Buscamos el filtro general para todas las imágenes sin obtener un resultado satisfactorio es decir un filtro que comprima mejor que los ya conocidos.

k) Los filtros generales, es decir por conjunto de imágenes no funcionan bien en la práctica.

l) Los filtros locales, por imagen, por contorno o por nivel de resolución ofrecen resultados muy satisfactorios.

m) Una de las principales características que ofrece el método LM es que no depende del operador decimación que tomemos.

n) Hemos obtenidos buenos resultados utilizando el método LM para funciones oscilatorias.

ñ) Hemos redefinido las características de la multiresolución para el caso del aprendizaje y hemos dado las bases para construir esquemas estables.

o) Hemos utilizado distintas normas en el funcional a minimizar recurriendo a técnicas de programación lineal para solucionar los diferentes problemas.

p) Los resultados obtenidos utilizando tanto métodos de núcleo como métodos LM mejoran los resultados obtenidos con métodos clásicos.

Futuras líneas de investigación

Esquematizamos las líneas de investigación que podemos considerar:

a) Podemos generalizar a dos dimensiones la construcción de un operador monótono de Hermite utilizando ENO y WENO para su posible utilización en reconstrucciones faciales por ordenador.

b) Podemos generalizar la estrategia (AY) a distintos operadores decimación.

c) La estrategia (AY) nos permite generalizar los métodos basados en medias PPH para otros operadores decimación distintos a valores puntuales.

d) Se podrían estudiar algunas características de los métodos de núcleo que repercutan directamente en la multiresolución, como la búsqueda de una función peso más concentrada en su centro y con una bajada más pronunciada.

e) En la generalización de los métodos de núcleo a medias en celda se ha definido una pseudodistancia, se podrían ver las consecuencias de utilizar otras como la distancia de Hausdorff.

f) Se podría estudiar las propiedades teóricas al cambiar la norma 2 por la norma p en los métodos de núcleo.

g) Se podría trabajar tanto en métodos de núcleo como en LM cambiando la clase de funciones a minimizar $\mathcal{K}$.

h) Podríamos cambiar el problema de programación lineal planteado (que consiste en minimizar la norma de los errores) por otros mejor adaptados.

i) Se pueden utilizar otras técnicas dentro del campo de teoría estadística de aprendizaje como métodos de árbol o redes neuronales para realizar multiresolución.

Bibliografía

[1] R. Acar and C. R. Vogel. Analysis of total variation penalty methods for ill-possed prolbmes. *Invers. Prob.*, 10:1212–1299, 1994.

[2] S. Amat, F. Aràndiga, A. Cohen, and R. Donat. Tensor product multiresolution analysis whith error control for compact representation. *Signal Processing*, 82:587–608, 2002.

[3] S. Amat, F. Aràndiga, A. Cohen, R. Donat, G. Garcia, and M. Von Oehsen. Data compression with ENO schemes - a case study. *Appl. Comput. Harmon. Anal.*, 11:273–288, 2002.

[4] S. Amat, R. Donat, J. Liandrat, and J. C. Trillo. Analysis of a new nonlinear subdivision scheme. *Applications in Image Processing, Found. Comp. Maht*, 6:193–225, 2006.

[5] S. Amat, R. Donat, J. Liandrat, and J. C. Trillo. A fully adaptive PPH multi-resolution scheme for image processing. *Math. Comput. Modelling*, 46:2–11, 2007.

[6] T. W. Anderson. *The Statistical Analysis of Time Series*. Wiley, New York, 1971.

[7] F. Aràndiga. Interpolatory data compression algorithms with error control. Technical report, Grupo ANIMS, Departamento de Matemática Aplicada, U. Valencia, 2003.

[8] F. Aràndiga, J. Baccou, M. Doblas, and J. Liandrat. Image compression based on a multi-directional map-dependent algorithm. *Appl. Comput. Harmon. Anal.*, 21:181–297, 2007.

[9] F. Aràndiga, A. Baeza, and D. F. Yáñez. Monotone Hermite interpolation based on ENO and WENO schemes. *En preparación*, 2009.

[10] F. Aràndiga, A. Cohen, P. Mulet, and D. F. Yáñez. Learning-based Multiresolution with ℓ^1-norm based on the lagged diffusivity fixed point method. *En preparación*, 2009.

[11] F. Aràndiga, A. Cohen, and D. F. Yáñez. Learning-based multiresolution schemes: Application to compression of images. *En preparación*, 2009.

[12] F. Aràndiga and R. Donat. Multi-scale decompositions by decimation and prediction. Technical report, Grupo ANIMS, Departamento de Matemática Aplicada, U. Valencia, 2000.

[13] F. Aràndiga and R. Donat. Nonlinear multiscale descompositions: The approach of A. Harten. *Numerical Algorithms*, 23:175–216, 2000.

[14] F. Aràndiga, R. Donat, A. Cohen, N. Dyn, and B. Mateï. Approximation of piecewise smooth functions and images by edge-adapted (ENO-EA) nonlinear multiresolution techniques. *Appl. Comput. Harmon. Anal.*, 24:225–250, 2008.

[15] F. Aràndiga, R. Donat, and A. Harten. Multiresolution based on weighted averages of the hat function I: linear reconstruction technique. *SIAM J. Numer. Anal.*, 36:160–203, 1998.

[16] F. Aràndiga, R. Donat, and A. Harten. Multiresolution based on weighted averages of the hat function II: Nonlinear reconstruction technique. *SIAM J. Numer. Anal.*, 20:1053–1093, 1999.

[17] F. Aràndiga, A. M. Belda, and P. Mulet. Point-value WENO multiresolution applications to stable image compression. *Journal of Scientific Computing*, 43(2):158–182, 2010.

[18] A. M. Belda. *Técnicas de Interpolación WENO y su aplicación al procesamiento de imágenes*. PhD thesis, Universitat de València, 2010.

[19] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[20] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, 2004.

[21] C. Brézinski. *Introduction à la pratique du calcul numérique*. Dunod Université, Paris, 1988.

[22] R. Burden, J. Faires, and A. Reynolds. *Numerical analysis*. PWS Publisheres, Boston, 1981.

[23] P. J. Burt and E. H. Adelson. The Laplacian pyramid as a compact image code. *IEEE Trans. Comm.*, 31:532–540, 1983.

[24] J. F. Canny. Finding edges and lines in images. Technical report, MIT Artificial Intelligence Laboratory, 1983.

[25] T. F. Chan, G. H. Golub, and P. Mulet. A nonlinear primal-dual method for total variation-based image restoration. *SIAM J. Numer. Anal.*, 20:1964–1977, 1997.

[26] T. F. Chan and P. Mulet. On the convergence of the lagged diffusivity fixed point method in total variation image restoration. *SIAM J. Numer. Anal.*, 36:354–367, 1998.

[27] B. Cheng and D. M. Titterington. Neural networks: A review from a statistical perspective (with discussion). *Stat. Science*, 9:2–54, 1994.

[28] R. Claypoole, G. Davis, W. Sweldens, and R. Baraniuk. Nonlinear wavelet transforms for image coding. *Proceedings of the 31st IEEE Asilomar Conference, Pacific Grove, CA,*, 1997.

[29] W. S. Cleveland and S. J. Devlin. Locally weighted regression: An approach to regression analysis by local fitting. *J. of the Amer. Stat. Association*, 83:596–610, 1988.

[30] A. Cohen, I. Daubechies, and J. Feauveau. Biorthogonal bases of compactly supported wavelets. *Comm. Pure Appl. Math.*, 45:485–560, 1992.

[31] A. Cohen, N. Dyn, and B. Mateï. Quasilinear subdivision schemes with applications to ENO interpolation. *Appl. Comput. Harmon. Anal.*, 15:89–116, 2003.

[32] S. Conte and C. De Boor. *Análisis numérico*. McGraw Hill, Mèxic D.F., 1974.

[33] K. Dadourian. *Schemes de subdivision, analyses multiresolutions non-linearies. Applications*. PhD thesis, Université de Provence, 2008.

[34] I. Daubechies. *Ten lectures on wavelets*. SIAM, Philadelphia, PA, 1992.

[35] P. Davis. *Interpolation & approximation*. Dover Publication, New York, 1975.

[36] C. De Boor and B. Swartz. Piecewise monotone interpolation. *J. Approx. Theory*, 21:411–416, 1977.

[37] E. L. De Forest. On some methods of interpolation applicable to the graduation of irregular series, such as tables of mortality. *Annual Report of the Board of Regents of the Smithsonian Institution for 1873*, 1873.

[38] M. Doblas. *Técnicas Interpolatorias no Lineales y Aplicaciones*. PhD thesis, Universitat de València, 2010.

[39] D. L. Donoho. Interpolating wavelet transforms, 1992.

[40] L. Eden and L. Wittmeyer. *Numerical Analysis*. Academic Press, New York, 1990.

[41] I. Ekeland and R. Temam. *Analyse convexe et problèmes variationnels*. Dunod, Paris, 1974.

[42] V. K. Epanechnikov. Nonparametric estimation of a multivariate probability density. *Theory of Probability and its Applications*, 14:153–158, 1969.

[43] J. Fan and I. Gijbels. *Local Polynomials Modelling and its Applications*. Chapman and Hall, London, 1996.

[44] F. N. Fritsch and J. Butland. A method for constructing local monotone piecewise cubic interpolants. *SIAM J. Sci. Stat. Comput.*, 5, 2:300–304, 1984.

[45] F. N. Fritsch and R. E. Carlson. Monotone piecewise cubic interpolation. *SIAM J. Numer. Anal.*, 17, 2:238–246, 1980.

[46] P. Getreuer and F. Meyer. ENO multiresolution schemes with general discretizations. *SIAM J. Numer. Anal.*, 46:2953–2977, 2008.

[47] I. Gijbels, A. Lambert, and P. Qiu. Jump-Preserving Regression and Smoothing using Local Linear Fitting: A Compromise. *Annals of the Institute of Statistical Mathematics.*, 59, 2:235–272, 2007.

[48] J. P. Gram. *Om Raekkeuviklinger, bestendeved Hjaelp af de mindste Kvadraters Methode*. A. F. Host & Son, Copenhagen, 1879.

[49] M. Grant and S. Boyd. Graph implementations for nonsmooth convex programs. *Recent Advances in Learning and Control (a tribute to M. Vidyasagar), V. Blondel, S. Boyd, and H. Kimura, editors, pages 95-110, Lecture Notes in Control and Information Sciences, Springer*, 2008.

[50] M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming (web page and software). *http://stanford.edu/ boyd/cvx*, 2009.

[51] A. Harten. ENO schemes with subcell resolution. *J. Comput. Phys.*, 83:148–184, 1989.

[52] A. Harten. Discrete multiresolution analysis and generalized wavelets. *J. Appl. Numer. Math.*, 12:153–192, 1993.

[53] A. Harten. Multiresolution representation of data: General framework. *SIAM J. Numer. Anal.*, 33:1205–1256, 1996.

[54] A. Harten, B. Engquist, S. Osher, and S. Chakravarthy. Uniformly high order accurate essentially non-oscillatory schemes III. *J. Comput. Phys.*, 71:231–303, 1987.

[55] T. Hastie and R. Tibshirani. *Generalized Additive Models*. Chapman and Hall, London, 1990.

[56] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, New York, 2001.

[57] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference and Prediction. Second Edition*. Springer, New York, 2009.

[58] X. C. He and N. H. C. Yung. Corner detector based on global and local curvature properties. *Optical Engineering*, 47(5):057008, 2008.

[59] H. Heijmans and J. Goutsias. Nonlinear multiresolution signal decomposition schemes- Part II: Morphological wavelets. *IEEE Trans. Image Process.*, 9:1897–1913, 2000.

[60] R. Henderson. Note on graduation by adjusted average. *Trans. of the Actuarial Society of America*, 17:43–48, 1916.

[61] P. J. Huber. Robust estimation of a location parameter. *Annals of Math. Stat.*, 53:73–101, 1964.

[62] P. J. Huber. *Robust Statistics*. Willey, 1981.

[63] J. H. Hyman. Accurate monotonicity preserving cubic interpolation. *SIAM J. Numer. Anal.*, 4, 4:645–654, 1983.

[64] S. Jaffard, Y. Meyer, and R. D. Ryan. *Wavelets, Tools for science & Technology*. SIAM, Philadelphia, PA, 2001.

[65] G-S. Jiang and C.-W. Shu. Efficient implementation of weighted ENO schemes. *J. Comput. Phys.*, 83:202–228, 1996.

[66] V. Y. Katkovic. Linear and nonlinear methods of nonparametric regression analysis. *Soviet Automatic Control*, 5:35–46, 1979.

[67] M. Kendall and J. K. Ord. *Time Series*. Oxford University Press, New York, 1990.

[68] K. Kohlmann. Corner detection in natural images based on the 2-D hilbert transform. *Signal Processing*, 48:225–234, 1996.

[69] X-D. Liu, S. Osher, and T. Chan. Weighted Essentially Non-oscillatory Schemes. *J. of Comp. Phys.*, 115:202–212, 1994.

[70] C. Loader. *Local Regression and likelihood*. Springer, New York, 1999.

[71] F. R. Macaulay. *Smoothing of Time Series*. National Bureau of Economic Research, New York, 1931.

[72] S. Mallat. *A Wavelet Tour of signal processing*. Academic Press, New York, 1999.

[73] B. Mateï. *Methodes Multiresoltuions non-linéaires applications au traitement d'image*. PhD thesis, Université P. et M. Curie, Paris, 2002.

[74] J. A. McDonald and Owen A. B. Smoothing with split linear fits. *Technometrics*, 28:195–208, 1986.

[75] D. H. McLain. Drawing contours from arbitrary data. *J. Assoc. Comput. Jour.*, 17:318–324, 1974.

[76] A. Peyronne. Utilisation de méthodes adaptatives pur l'angrandissement d'images. Technical report, Grupo ANIMS, Departamento de Matemática Aplicada, U. Valencia, 2001.

[77] J. M. S. Prewitt. Object enhancement and extraction. *B.S. Lipkin and A. Rosenfeld, Editors, Picture Processing and Psychopictorics, Academic Press, New York*, 1970.

[78] B. D. Ripley. *Pattern Recognition and Neural Networks*. Cambridge, University Press, 1996.

[79] G. V. Schiaparelli. Sul modo di ricavare la vera espressione delle leggi delta natura dalle curva empiricae. *Effemeridi Astronomiche di Milano per l'Arno*, 857:3–56, 1866.

[80] J. Shiskins, A. H. Young, and J. C. Musgrave. The X-11 variant of the census method II seasonal adjustment program. *Technical Report, Bureau of the Census, U. S. Departament of Commerce*, 1967.

[81] I. Sobel and G. Feldman. A 3x3 Isotropic Gradient Operator for Image Processing. *Presented at the Stanford Artificial Project*, 1968.

[82] J. Spencer. On the graduation of rates of sickness and mortality. *Journal of the Institute of Actuaries*, 38:334–343, 1904.

[83] J. Stoer and R. Bulirsch. *Introduction to Numerical Analysis*. Springer Verlag, New York, 1979.

[84] C. J. Stone. Consistent nonparametric regression. *The Annals of Statistics*, 5:595–645, 1977.

[85] C. J. Stone. Optimal rates of convergence for nonparametric estimators. *The Annals of Statistics*, 8:1348–1360, 1980.

[86] C. J. Stone. Optimal rates of convergence for nonparametric regression. *The Annals of Statistics*, 10:1040–1053, 1982.

[87] G. Strang. *Introduction to Applied Mathematics*. Cambridge Press, Wellesley, 1986.

[88] W. Sweldens. The lifting scheme: A construction of second generation wavelets. *SIAM J. Numer. Anal.*, 29:511–546, 1998.

[89] S. Thurnhofer, M. Lighstone, and S. K. Mitra. Adaptative interpolation of images with application to interlaced-to-progressive conversion. *Proc. SPIE Symp. Visual Communications and Image Processing*, pages 614–625, 1993.

[90] V. N. Vapnik. *The Nature of Statistical Learning*. Springer, New York, 1995.

[91] C. R. Vogel and M. E. Oman. Iterative methods for total variation denoising. *SIAM J. Sci.*, 25:243–251, 1996.

[92] M. P. Wand and M. C. Jones. *Kernel Smoothing*. Chapman and Hall, London, 1995.

[93] G. Wolberg and I. Alfy. Monotonic cubic spline interpolation. In *CGI '99: Proceedings of the International Conference on Computer Graphics*, page 188, Washington, DC, USA, 1999. IEEE Computer Society.

[94] D. F. Yáñez. *Compresión de imágenes digitales. Learning Schemes*. Trabajo de Investigación, Grupo ANIMS, U. Valencia, 2007.