

## FILTER BANK METHODS FOR HYPERBOLIC PDEs\*

JOHAN WALDÉN†

**Abstract.** We use biorthogonal filter banks to solve hyperbolic PDEs adaptively with a sparse multilevel representation of the solution. The methods described are of finite difference type, and the filter banks are used to give a sparse representation of signals and to transform between grids on different scales. We derive bounds for the error and number of coefficients in the sparse representation. These bounds also apply for filter banks that are not associated with any wavelets. We develop algorithms for fast differentiation and multiplication in detail. The strength of the method is shown in various test problems.

**Key words.** wavelets, adaptive PDE methods, hyperbolic equations, numerical algorithms

**AMS subject classifications.** 65D25, 65M06, 65M50, 65M55

**PII.** S0036142996313130

**1. Introduction.** We wish to solve time dependent initial value problems of hyperbolic type

$$(1.1) \quad \begin{cases} u_t = Pu, \\ u(x, 0) = u_0(x), \\ x \in \mathbb{R}^n, t \geq 0, \end{cases}$$

where the solution is smooth in large parts of the domain but has small regions of sharp gradients. For such problems wavelets offer a way of solving the equation adaptively. This was used in [18] to develop a solver working in the wavelet domain, based on a Galerkin method. The method used orthonormal Daubechies wavelets and worked satisfactorily for PDEs with constant coefficients. Unfortunately, it was not efficient for multiplication with nonconstant functions, a problem that seems hard to avoid when working purely in the wavelet domain. This has also been noticed elsewhere, e.g., in [16, 20, 30].

On the contrary, multiplication is easy in the domain of scaling coefficients. Therefore, we need a method that works in both the wavelet and scaling coefficient domains. Multiplication will be especially simple when the scaling coefficients can be viewed as point values. This was used in [4, 5] with coiflets and in [16, 27, 30, 31] with collocation methods. When a collocation method is used, a problem is that a convolution with a filter of infinite length (but with coefficients that decrease fast) is needed, even if the wavelets have compact support. We refer to these methods as “pure” wavelet methods, as the solution is represented as a wavelet decomposition of  $L^2$  (or some other space).

Another approach is to use the wavelet transform in an adaptive method to “flag” where the resolution needs to be increased, and use some other method for the actual solver. This idea was used in [20, 21], where finite difference methods were developed, using only point value representations of functions but using Daubechies wavelets to refine grids adaptively. In [17] such a “flagging technique” was used together

\*Received by the editors December 4, 1996; accepted for publication (in revised form) July 28, 1998; published electronically June 15, 1999. This work was supported by the Swedish Natural Science Research Council, NFR, under grant F-FU 04370–304.

<http://www.siam.org/journals/sinum/36-4/31313.html>

†Department of Scientific Computing, Uppsala University, P.O. Box 120, SE-751 04 Uppsala, Sweden (johan@tdb.uu.se).

with total variation diminishing (TVD) methods for solving conservation laws. The idea was to avoid evaluation of fluxes wherever possible by interpolating from coarse representations.

In this paper we develop the *filter bank method*, which lies somewhere between the “pure approach” and the “flagging technique.” The filter bank method uses a pointwise multilevel representation of the solution, and biorthogonal filter banks that do not necessarily correspond to wavelets are used to transform between different levels of this representation. We will borrow notation from signal analysis where perfect reconstruction filter banks were originally introduced [2, 26]. The use of such filter banks will be justified theoretically. Differential operators will be approximated with finite difference operators, which leads to shorter filters than obtained when using Galerkin or collocation methods. The analysis in this paper for which filter banks should be used shows that neither the filter banks used in [4, 5] (coiflets) nor the ones in [20, 21] (Daubechies filters) are optimal.

The filter bank method is designed to avoid the problems that appear when using wavelet decompositions of  $L^2$  to solve hyperbolic PDEs (for example, inefficient pointwise multiplication, long stencils for differential operators, and the need for modification of many basis functions close to boundaries). The method could be used to add adaptivity, in a fairly general way, to a finite difference solver that works on a uniform Cartesian grid. This comes from the fact that although the representation of the solution is *nonuniform*, all operators are applied to locally *uniform* grids when using the filter bank method.

As the equations in this paper have periodic boundary conditions (BCs), the filter bank method is still an “experimental method.” However, the combination of working (partly) in the physical space and with filters of short support implies that adding BCs ought not to be a major difficulty. In a forthcoming paper, we will show results for problems with BCs, and we will also measure the speedup in CPU time that can be achieved compared with a nonsparse solver.

Compared with the methods developed in [20, 21], there are some advantages in using only uniform grids for operators. As already mentioned, this is a fairly simple way to add adaptivity to a solver that works on a uniform grid. Furthermore, the method is easily generalized to higher dimensions, and it might be more stable than a nonuniform solver. Also, using the  $\delta$ -filter banks instead of the Daubechies filter banks has, as already mentioned, the advantage of permitting simpler treatment of BCs and pointwise multiplication. The problems solved in this paper all have smooth solutions (but with large gradients), so there is no need to use the TVD approach as done in [17]. However, in future work the performance of the filter bank method for conservation laws will be analyzed.

There are of course some *disadvantages* in using the filter bank method. As it is based on finite difference methods there is, as always, a question of stability. Furthermore, compared with methods that use overlapping grids or domain decomposition there are restrictions in that we always use a subdivision factor of two, and that the orientation of the different grids are the same. These restrictions seem difficult to overcome within the framework for perfect reconstruction filter banks. However, in the numerical examples of this paper, no stability problems have occurred (as long as the underlying finite difference solver is stable), and this robustness of the method might be an advantage compared with more general subspace decompositions.

With our approach, the wavelet transform is used to perform what are its strengths:

- It detects local high frequencies.

- It gives compact representations of functions with small regions of large gradients.
- It provides a “rule” of how to go from a coarse scale representation to a finer scale.

The paper consists of the following parts. In section 2 we introduce some preliminary notation on wavelets and filter banks. In section 3 we introduce the truncated wavelet representation and compare the filter bank method with the wavelet Galerkin method. We also present fast algorithms for differentiating a function and multiplying two functions in the wavelet domain. These algorithms are presented in detail and we call them  $\Lambda$ -cycles as they, in contrast to the multigrid  $V$ -cycle, go from coarse scales to fine. Finally, in sections 4 and 5 we study a number of linear and nonlinear test problems. The proposed method is shown to perform well, with compression factors of up to 110.

**2. Preliminaries.** Although we will be concerned mainly with the wavelet transform on  $\mathbb{Z}$  we start with the continuous theory. We will use the Hilbert space  $L^2(\mathbb{R})$ , with the inner product

$$(2.1) \quad \langle f, g \rangle = \int_{-\infty}^{\infty} f \bar{g} \, dx,$$

and the Banach spaces  $L^p(\mathbb{R})$ ,  $L^\infty(\mathbb{R})$ , with norms

$$(2.2) \quad \|f\|_p = \left( \int_{-\infty}^{\infty} |f|^p \, dx \right)^{1/p}, \quad p \in [1, \infty),$$

$$(2.3) \quad \|f\|_\infty = \operatorname{esssup}_x |f(x)|.$$

The Hölder spaces  $C^\gamma$  are defined by

$$(2.4) \quad \begin{aligned} f \in C^\gamma &\Leftrightarrow \sup_{x,y} \frac{|f(x) - f(y)|}{|x - y|^\gamma} < \infty, \quad \gamma \in (0, 1], \\ f' \in C^\gamma &\Leftrightarrow f \in C^{\gamma+1}. \end{aligned}$$

(This is a slight abuse of notation as, for  $\gamma \in \mathbb{N}$ ,  $C^\gamma$  is actually the Lipschitz space,  $Lip_\gamma$ .) We have the seminorms on  $C^\gamma$ ,

$$(2.5) \quad \|f\|_{\dot{C}^\gamma} = \sup_{x,y} \frac{|f^{(\lfloor \gamma \rfloor)}(x) - f^{(\lfloor \gamma \rfloor)}(y)|}{|x - y|^{\gamma - \lfloor \gamma \rfloor}},$$

where  $\lfloor \gamma \rfloor$  is defined as the largest integer not larger than  $\gamma$ .

We define the Fourier transform of a function  $f \in L^1(\mathbb{R})$  by

$$(2.6) \quad \hat{f}(\xi) = \int f(x) e^{-i\xi x} \, dx,$$

and we will use the standard generalizations to other spaces. The Sobolev spaces,  $H^s(\mathbb{R})$ ,  $s \in \mathbb{R}$ , are Hilbert spaces with inner products

$$(2.7) \quad \langle f, g \rangle_{H^s} = \frac{1}{2\pi} \int \hat{f} \bar{\hat{g}} (1 + \omega^2)^s \, d\omega.$$

Downsampling of a function  $\downarrow$  is defined as  $(\downarrow f)(x) = f(2x)$  and upsampling as  $(\uparrow f)(x) = f(x/2)$ . By  $\mathcal{P}^k$ , we mean the set of polynomials of degree at most  $k$ . We define  $\mathbb{T}$  as  $\mathbb{R}/\mathbb{Z}$  (and *not* as  $\mathbb{R}/2\pi\mathbb{Z}$  which is more common).

For the discrete theory we will borrow some notation from signal analysis. A filter (discrete function, grid function) is a function  $u : \mathbb{Z} \rightarrow \mathbb{C}$ . We will sometimes use the notation  $u_k$  for  $u(k)$ . Some subspaces of the space of all filters  $\mathbb{C}^{\mathbb{Z}}$  are the Banach spaces  $l^p$  with norms

$$(2.8) \quad \|u\|_p = \left( \sum_k |u(k)|^p \right)^{1/p}, \quad p \in [1, \infty),$$

and the Hilbert space  $l^2$  with inner product

$$(2.9) \quad \langle u, v \rangle = \sum_k u(k) \overline{v(k)}.$$

We will use the mirror operator  $\check{u}(k) = u(-k)$ . The convolution of two filters is  $(f * g)(k) = \sum_n f(n)g(k-n)$  (when the right-hand side is defined). Downsampling of a filter is defined as  $(\downarrow f)(k) = f(2k)$  and upsampling as  $(\uparrow f)(k) = f(k/2)$  when  $k$  is even, and  $(\uparrow f)(k) = 0$  when  $k$  is odd. In signal analysis these operations are often denoted  $\downarrow 2$  and  $\uparrow 2$ , but as we will work only with a subsampling factor of 2, we choose the shorter notation.

We define the discrete Fourier transform, acting on grid functions (in  $l^1$ ) and with an image consisting of  $2\pi$ -periodic functions by

$$(2.10) \quad \hat{u}(\omega) = \sum_k u_k e^{-i\omega k}.$$

The inverse transform will then be

$$(2.11) \quad u(k) = \frac{1}{2\pi} \int_{-\pi}^{\pi} \hat{u}(\omega) e^{i\omega k} d\omega,$$

and we have

$$\begin{aligned} \widehat{u * v} &= \hat{u} \hat{v}, \\ \widehat{(\uparrow u)}(\omega) &= \hat{u}(2\omega), \\ \widehat{(\downarrow u)}(\omega) &= \frac{1}{2}(\hat{u}(\omega/2) + \hat{u}(\omega/2 + \pi)). \end{aligned}$$

The convolution of  $u$  and  $v$  can be seen as applying a linear operator on  $v$ . A general linear operator on a grid function  $A$  can be represented by an infinite matrix  $\mathbf{A} \in \mathbb{C}^{\mathbb{Z} \times \mathbb{Z}}$ , and the case of convolution arises when  $A$  is a Toeplitz operator, i.e., when  $\mathbf{A}_{i,j} = \mathbf{A}_{i-j,0} \ \forall i \ \forall j$ . For linear operators we define the operator norms

$$(2.12) \quad \|A\|_p = \sup_u \frac{\|Au\|_p}{\|u\|_p}.$$

When combining different operators on filters we use a right-to-left rule, for example,  $\downarrow u * v = \downarrow (u * v)$ . We will sometimes write  $hg$  instead of  $h * g$  (interpreting  $h$  as the Toeplitz operator acting on  $g$ ). The adjoint,  $A^*$  of a linear operator on  $l^2$ , is the operator for which  $\langle Au, v \rangle = \langle u, A^*v \rangle \ \forall u \ \forall v$ . We have the following relations:

$$\begin{aligned} (AB)^* &= B^*A^*, \\ \downarrow^* &= \uparrow, \\ h^* &= \check{h}, \end{aligned}$$

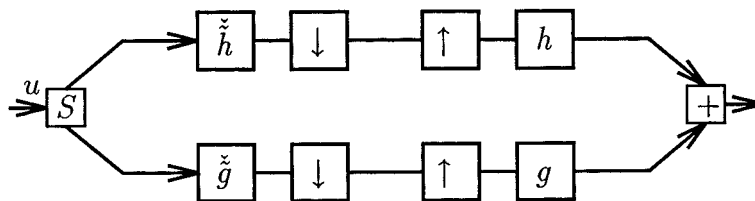


FIG. 1. Block diagram notation for operations on grid functions.

leading to  $(\downarrow \check{h})^* = h \uparrow$ .

The support of a filter is naturally defined as  $\text{supp } u = \{k : u_k \neq 0\}$ . We will use the lower bound  $m(u) = \inf \text{supp } u$ , the upper bound  $M(u) = \sup \text{supp } u$ , and the length of a filter,  $\text{length } u = M(u) - m(u) + 1$ . An easy way to display a filter with finite support is on the form

$$u = (u_{m(u)}, u_{m(u)+1}, \dots, u_{-1}, \underline{u}_0, u_1, \dots, u_{M(u)-1}, u_{M(u)}),$$

where the zeroth coordinate of  $u$  is underlined. The discrete ordered interval  $[a, b]_{\mathbb{Z}}$  is defined as  $\{a, a+1, \dots, b-1, b\}$ .

We will go from functions on  $\mathbb{R}$  to  $\mathbb{Z}$  with the sampling operator  $[\ ]_h : \mathbb{C}^{\mathbb{R}} \rightarrow \mathbb{C}^{\mathbb{Z}}$ , which is defined through  $[f]_h(k) = f(kh)$ . We will also use the convolution of a filter and a function on  $\mathbb{R}$ ,  $(u * f)(x) = \sum_k u_k f(x - k)$ . A nice way to show operations on a filter is with a block diagram. For example, in Figure 1 the operation  $h \uparrow \downarrow \check{h} u + g \uparrow \downarrow \check{g} u$  is shown. (The operation  $S$  stands for “split.”)

We now turn to the continuous wavelet transform by introducing the biorthogonal multiresolution analysis (MRA) [8, 24]. An MRA consists of two sequences of closed subspaces,  $V_j, \tilde{V}_j$  of  $L^2$ , satisfying

$$(2.13) \quad \begin{aligned} \cdots \subset V_{-2} \subset V_{-1} \subset V_0 \subset V_1 \subset V_2 \subset \cdots, \\ \cdots \subset \tilde{V}_{-2} \subset \tilde{V}_{-1} \subset \tilde{V}_0 \subset \tilde{V}_1 \subset \tilde{V}_2 \subset \cdots, \end{aligned}$$

$$(2.14) \quad \overline{\bigcup_{j \in \mathbb{Z}} V_j} = \overline{\bigcup_{j \in \mathbb{Z}} \tilde{V}_j} = L^2(\mathbb{R}),$$

$$(2.15) \quad \bigcap_{j \in \mathbb{Z}} V_j = \bigcap_{j \in \mathbb{Z}} \tilde{V}_j = \{0\},$$

$$(2.16) \quad \begin{aligned} f \in V_j &\Leftrightarrow f(2^{-j} \cdot) \in V_0, \\ f \in \tilde{V}_j &\Leftrightarrow f(2^{-j} \cdot) \in \tilde{V}_0, \end{aligned}$$

$$(2.17) \quad \begin{aligned} f \in V_0 &\Leftrightarrow f(\cdot - k) \in V_0, \\ f \in \tilde{V}_0 &\Leftrightarrow f(\cdot - k) \in \tilde{V}_0, \end{aligned}$$

$$\begin{aligned}
(2.18) \quad & \exists \varphi, \exists \tilde{\varphi} : \{\varphi(\cdot - k)\}_k \text{ is a Riesz basis of } V_0, \\
& \{\tilde{\varphi}(\cdot - k)\}_k \text{ is a Riesz basis of } \tilde{V}_0, \\
& \langle \varphi(\cdot - k), \tilde{\varphi} \rangle = \delta_k.
\end{aligned}$$

We call  $\varphi, \tilde{\varphi}$  the *scaling functions*. Throughout the paper we will assume that these are real. With this construction we have real filters,  $h, \tilde{h}$ , such that

$$\begin{aligned}
(2.19) \quad & \varphi(x) = \sqrt{2} \sum_k h_k \varphi(2x - k), \\
& \tilde{\varphi}(x) = \sqrt{2} \sum_k \tilde{h}_k \tilde{\varphi}(2x - k),
\end{aligned}$$

and we define

$$\begin{aligned}
(2.20) \quad & \varphi_{jk}(x) = 2^{j/2} \varphi(2^j x - k), \\
& \tilde{\varphi}_{jk}(x) = 2^{j/2} \tilde{\varphi}(2^j x - k).
\end{aligned}$$

By defining the *mother wavelets*

$$\begin{aligned}
(2.21) \quad & \psi(x) = \sqrt{2} \sum_k g_k \varphi(2x - k), \\
& \tilde{\psi}(x) = \sqrt{2} \sum_k \tilde{g}_k \tilde{\varphi}(2x - k),
\end{aligned}$$

where

$$\begin{aligned}
(2.22) \quad & g_k = (-1)^k \tilde{h}_{1-k}, \\
& \tilde{g}_k = (-1)^k h_{1-k},
\end{aligned}$$

the wavelets

$$\begin{aligned}
(2.23) \quad & \psi_{jk}(x) = 2^{j/2} \psi(2^j x - k), \\
& \tilde{\psi}_{jk}(x) = 2^{j/2} \tilde{\psi}(2^j x - k),
\end{aligned}$$

and the spaces  $W_j = \overline{\text{span}\{\psi_{jk}\}_k}$ ,  $\tilde{W}_j = \overline{\text{span}\{\tilde{\psi}_{jk}\}_k}$ , we get biorthogonal bases of  $L^2$ , i.e., all  $\psi_{j,k}$  and all  $\tilde{\psi}_{j,k}$  are uniformly linearly independent, and

$$(2.24) \quad f = \sum_{j,k} \langle f, \psi_{jk} \rangle \psi_{jk} = \sum_{j,k} \langle f, \tilde{\psi}_{jk} \rangle \tilde{\psi}_{jk}.$$

We will always use the second of these expansions.

Our numerical results will be for problems with periodic boundary conditions. One way to define wavelets on the circle  $\mathbb{T}$  is to use the following definitions:

$$(2.25) \quad \varphi_{jk}^{\text{per}}(x) = \sum_l \varphi_{jk}(x + l),$$

$$(2.26) \quad \psi_{jk}^{\text{per}}(x) = \sum_l \psi_{jk}(x + l),$$

generating the spaces

$$(2.27) \quad V_j^{\text{per}} = \text{span}(\{\varphi_{jk}^{\text{per}}(x)\}_{k=0,\dots,2^j-1}),$$

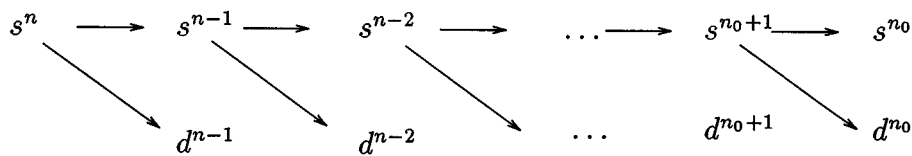


FIG. 2. The pyramid fast wavelet transform.

$$(2.28) \quad W_j^{\text{per}} = \text{span}(\{\psi_{jk}^{\text{per}}(x)\}_{k=0,\dots,2^j-1}),$$

and similarly for  $\tilde{\varphi}, \tilde{\psi}$ .

The biorthogonality relation (2.18) implies that the filters  $h, \tilde{h}$  satisfy the relation

$$(2.29) \quad \sum_n h_n \tilde{h}_{n+2k} = \delta_k,$$

and the dilation equation (2.19) implies that we have the condition

$$(2.30) \quad \sum_k h_k = \sum_k \tilde{h}_k = \sqrt{2}.$$

For a function,  $f \in V_j$ , with  $s_k^j = \langle f, \tilde{\varphi}_{jk} \rangle$ , the coefficients  $d_k^{j-1} = \langle f, \tilde{\psi}_{j-1,k} \rangle$ ,  $s_k^{j-1} = \langle f, \tilde{\varphi}_{j-1,k} \rangle$  can be found from (2.21),

$$(2.31) \quad \begin{aligned} s_k^{j-1} &= \sum_n \tilde{h}_{n-2k} s_n^j, \\ d_k^{j-1} &= \sum_n \tilde{g}_{n-2k} s_n^j. \end{aligned}$$

We can also go backward:

$$(2.32) \quad s_k^j = \sum_n (h_{k-2n} s_n^{j-1} + g_{k-2n} d_n^{j-1}).$$

Written in filter notation this becomes  $s^j = h \uparrow \tilde{h} s^j + g \uparrow \tilde{g} s^j$ , which is the operation in Figure 1. The same operation can of course be made recursively leading to the pyramid algorithm shown in Figure 2, which is an  $\mathcal{O}(N)$  algorithm, where  $N$  is the number of nonzero coefficients in  $s^j$ . Of course, when doing numerical calculations, we cannot deal with all scales  $j$ ; we have to choose a finest scale  $j_0$ . The Strang-Fix condition [15] implies that vanishing moments (zero-moments) for the wavelets guarantee a high order of approximation; i.e., if we define

$$(2.33) \quad \Sigma_n(\tilde{\psi}) = \int x^n \tilde{\psi}(x) dx,$$

then

$$(2.34) \quad \Sigma_n(\tilde{\psi}) = 0, \quad n = 0, \dots, N-1,$$

implies that

$$(2.35) \quad \|E_j f - f\|_2 = \mathcal{O}(2^{-jN}),$$

where  $E_j$  is the orthogonal projection operator of  $L^2$  onto  $V_j$ .

Not all choices of  $h, g, \tilde{h}, \tilde{g}$  satisfying (2.22), (2.29), (2.30) lead to dual wavelet bases. Extra conditions must be satisfied. Whereas for orthogonal filter banks the worst thing that can happen is that the  $\psi_{jk}$ 's constitute a tight frame of  $L^2$  instead of an orthonormal basis, when using nonorthogonal filter banks, blow-up of constants can appear, leading to  $\varphi$  or  $\tilde{\varphi} \notin L^2$ . For further discussions on this subtle topic, see [6, 7, 8, 10, 22, 23]. Conversely, from a filter point of view we can always choose filters  $h, g, \tilde{h}, \tilde{g}$  without worrying about whether they come from an MRA. We will see that such filters also will work well.

DEFINITION 2.1. A perfect reconstruction filter bank is a quadruple of filters  $(h, g, \tilde{h}, \tilde{g})$  satisfying

$$(2.36) \quad \sum_k h_k = \sum_k \tilde{h}_k = \sqrt{2},$$

$$(2.37) \quad \sum_n h_n \tilde{h}_{n+2k} = \delta_k,$$

$$(2.38) \quad \begin{aligned} g_k &= (-1)^k \tilde{h}_{1-k}, \\ \tilde{g}_k &= (-1)^k h_{1-k}. \end{aligned}$$

We will work only with real filters with a finite number of nonzero coefficients. If we define the  $n$ th discrete moment as

$$(2.39) \quad \sigma_n(h) = \sum_k k^n h_k,$$

the zero-moment conditions (2.34) carry over to discrete zero-moments on  $\tilde{g}$ :

$$(2.40) \quad \sigma_k(\tilde{g}) = 0, \quad k = 0, \dots, N-1.$$

The transform of a grid function  $s^n$  is now defined in the same manner as in the continuous case

$$(2.41) \quad \begin{aligned} \mathcal{F}^{n-n_0} s^n &= (d^{n-1}, d^{n-2}, \dots, d^{n_0}, s^{n_0}) = (t^{n-1}, t^{n-2}, \dots, t^{n_0-1}) = \tau, \\ d^{n-j} &= \tilde{G} \tilde{H}^{j-1} s^n, \\ s^{n_0} &= \tilde{H}^{n-n_0} s^n, \end{aligned}$$

and with a perfect reconstruction filter bank we get the inversion formula

$$(2.42) \quad s^n = (\mathcal{F}^{n-n_0})^{-1} \tau = \mathcal{F}^{n_0-n} \tau = \sum_{j=1}^{n-n_0} (H^*)^{j-1} G^* d^{n-j} + (H^*)^{n-n_0} s^{n_0}.$$

Here we have defined

$$(2.43) \quad \begin{aligned} \tilde{H} &= \frac{1}{\sqrt{2}} \downarrow \tilde{h}, & \tilde{G} &= \sqrt{2} \downarrow \tilde{g}, \\ H &= \sqrt{2} \downarrow h, & G &= \frac{1}{\sqrt{2}} \downarrow g. \end{aligned}$$

The factors,  $\sqrt{2}$ ,  $\frac{1}{\sqrt{2}}$ , appearing in the formulas will be justified when we introduce the  $\epsilon$ -truncated wavelet representation. They also come in handy for the filters we will use, as the coefficients will become rational numbers (with binary denominators). The superscript  $n$  in  $s^n$  naturally denotes the scale in the approximation, and we will sample  $s^n = [f]_{2^{-n}}$  in the subsequent sections. This transform, acting on point values, will be denoted the *filter bank transform* in this paper, whereas we will call



TABLE 2.1  
*Daubechies orthogonal filter banks ( $\tilde{h} = h$ ).*

Name	$N, \tilde{N}$	$h, \tilde{h}$
$D_4$	2	(0.48296291314, 0.836516303, 0.2241438680, -0.129409522)
$D_6$	3	(0.33267055295, 0.8068915093, 0.4598775021, -0.13501102001, -0.085441273, 0.035226291)
$D_8$	4	(0.2303778133, 0.7148465705, 0.63088076793, -0.0279837694, -0.18703481171, 0.0308413818, 0.032883011, -0.010597401)

TABLE 2.2  
*Spline filters ( $h$ ) with duals with short support.*

Name	$N$	$\tilde{N}$	$h$	$\tilde{h}$
$\beta_0$	1	1	$\frac{\sqrt{2}}{2}(1, 1)$	$\frac{\sqrt{2}}{2}(1, 1)$
$\beta_1$	2	1	$\frac{\sqrt{2}}{4}(1, 2, 1)$	$\frac{\sqrt{2}}{4}(-1, 2, 3)$
$\beta_2$	3	1	$\frac{\sqrt{2}}{8}(1, 3, 3, 1)$	$\frac{\sqrt{2}}{4}(-1, 3, 3, -1)$
$\beta_3$	4	1	$\frac{\sqrt{2}}{16}(1, 4, 6, 4, 1)$	$\frac{\sqrt{2}}{16}(3, -12, 10, 20, -5)$
$\beta_4$	5	1	$\frac{\sqrt{2}}{32}(1, 5, 10, 10, 5, 1)$	$\frac{\sqrt{2}}{16}(3, -15, 20, 20, -15, 3)$
$\beta_5$	6	1	$\frac{\sqrt{2}}{64}(1, 6, 15, 20, 15, 6, 1)$	$\frac{\sqrt{2}}{32}(-5, 30, -63, 28, 77, -42, 7)$
	3	3	$\frac{\sqrt{2}}{8}(1, 3, 3, 1)$	$\frac{\sqrt{2}}{64}(3, -9, -7, 45, 45, -7, -9, 3)$

TABLE 2.3  
*Filter banks with vanishing moments for  $h, \tilde{h}, g, \tilde{g}$ .*

Name	$N$	$\tilde{N}$	$h$	$\tilde{h}$
$\gamma_1$	2	2	$\frac{\sqrt{2}}{4}(1, 2, 1)$	$\frac{\sqrt{2}}{8}(-1, 2, 6, 2, -1)$
$\gamma_2$	3	3	$\frac{\sqrt{2}}{16}(3, 8, 6, 0, -1)$	$\frac{\sqrt{2}}{128}(3, 0, -12, 24, 82, 48, -12, -8, 3)$
$\gamma_3$	4	4	$\frac{\sqrt{2}}{32}(-1, 0, 9, 16, 9, 0, -1)$	$\frac{\sqrt{2}}{512}(-1, 0, 18, -16, -63, 144, 348, 144, -63, -16, 18, 0, -1)$
$\gamma_4$	5	5	$\frac{\sqrt{2}}{256}(-5, 0, 60, 128, 90, 0, -20, 0, 3)$	$\frac{\sqrt{2}}{32768}(15, 0, -280, 0, 1380, -640, -3240, 7680, 20634, 11520, -3240, -2560, 1380, 384, -280, 0, 15)$

TABLE 2.4  
*Filter banks with vanishing moments for  $\tilde{g}, \tilde{h}$ .*

Name	$N$	$\tilde{N}$	$h$	$\tilde{h}$
$\delta_1$	2	0	$\frac{\sqrt{2}}{4}(1, 2, 1)$	$(\sqrt{2})$
$\delta_2$	3	0	$\frac{\sqrt{2}}{16}(3, 8, 6, 0, -1)$	$(\sqrt{2})$
$\delta_3$	4	0	$\frac{\sqrt{2}}{32}(-1, 0, 9, 16, 9, 0, -1)$	$(\sqrt{2})$
$\delta_4$	5	0	$\frac{\sqrt{2}}{256}(-5, 0, 60, 128, 90, 0, -20, 0, 3)$	$(\sqrt{2})$
$\delta_5$	6	0	$\frac{\sqrt{2}}{512}(3, 0, -25, 0, 150, 256, 150, 0, -25, 0, 3)$	$(\sqrt{2})$

the MRA interpretation the *wavelet transform*. Periodization of the filters is done by treating them as circulant, with length  $2^j$  on scale  $j$ . We will always assume that the filters in the perfect reconstruction filter bank have length smaller than the coarsest scale, length  $h$ , length  $\tilde{h} < 2^{n_0}$ .

Some examples of perfect reconstruction filter banks are shown in Tables 2.1–Table 2.4. Here  $\tilde{N}$  is the number of vanishing moments for  $g$ , and  $N$  is the number

of vanishing moments for  $\tilde{g}$ . Table 2.1 contains the Daubechies filters that lead to orthonormal wavelets [10] (so  $\tilde{h} = h$ ), whereas Table 2.2 contains spline filters  $h$  and the duals of short support that satisfy the perfect reconstruction identity. For the third-order filter we have listed also a dual with longer support that has an associated function  $\tilde{\varphi} \in L^2$ . The two different filters will be compared in the next section. In Table 2.3 we have listed some interesting filters that also satisfy zero-moment conditions for  $h, \tilde{h}$  (except for the zeroth moment  $\sigma_0$ ). They are the biorthogonal versions of coiflets [12], and they have fewer nonzero coefficients than these. They are interesting, as inner products can be approximated with point values for them, and sparse multiplication will be of high order, as we will see. It will be shown that it is only the number of zero-moments of  $\tilde{h}$  that is important. This means that the  $\delta$  filters in Table 2.4 will be “optimal,” as they have only zero-moments for these. For these filters, the  $\tilde{H}$  and  $G^*$  steps will be trivial. The  $\delta$ -filters of even order were originally introduced as interpolation refinement schemes [13]. They are connected to Daubechies filters by the equation  $h^{\delta_{2N-1}} = h^{D_{2N}} * \tilde{h}^{D_{2N}}$  [11]. This connection was used in [3] to develop a collocation method for elliptic problems.

**3. The filter bank method.** The most frequently used approach when solving PDEs with wavelet methods is to use the continuous theory and approximate the solution with a Galerkin method or a collocation method. To approximate inner products one can use quadrature formulas proposed in [29]. The simplest formulas of the same order as (2.35) will be a convolution with  $\tilde{r}$ , where  $\tilde{r} = 2^{-j/2}[\tilde{\varphi}]_1$ . Conversely, one could take the point values of the sampled function and transform these. Furthermore, in the reconstruction, one could approximate the value of the function in the dyadic points by the coefficients in the wavelet expansion, instead of evaluating the exact value which would correspond to convolution with  $r = 2^{j/2}[\varphi]_1$ . The approximations would be

$$(3.1) \quad \begin{aligned} \langle f, \tilde{\varphi}_{jk} \rangle &\approx 2^{-j/2} f(2^{-j}k), \\ \sum_k c_k \varphi_{jk}(2^{-j}s) &\approx 2^{j/2} c_s. \end{aligned}$$

The approximations are of first order, but as the filters involved make up a perfect reconstruction filter bank, they give an exact reconstruction of the sampled values. The approach is of course the same as the filter bank method. There is one case when (3.1) actually gives a high-order approximation even from the continuous point of view, and that is when we use scaling functions with vanishing moments (i.e., coiflets, or the biorthogonal wavelets in Table 2.3). This was used in [4, 5] to get a high-order method. The question is: What happens with the signal when we start manipulating it, e.g., truncating coefficients? It turns out, as we will see, that the filter bank approach performs better in this case too.

Another advantage with using filter banks instead of the underlying wavelets is that we can use filters that do not generate wavelets (for example,  $\tilde{h} = \frac{\sqrt{2}}{4}(-1, 3, 3, -1)$ ). This gives us the opportunity to use shorter filters. We now introduce the truncated transformed signal.

**3.1. Truncation of functions.** Suppose we transform a discrete function  $s^n$  which can be thought of as a sampling of a function  $s^n = [f]_{2^{-n}}$ ,  $k = n - n_0 - 1$  steps, giving  $\tau = \mathcal{F}^k s^n$ . We define the set of retained coefficients

$$(3.2) \quad I_n^\epsilon = \{(j, k) : n_0 \leq j < n, |t_k^j| > \epsilon\}$$

and the number of significant coefficients

$$(3.3) \quad N_s = |I_n^\epsilon|.$$

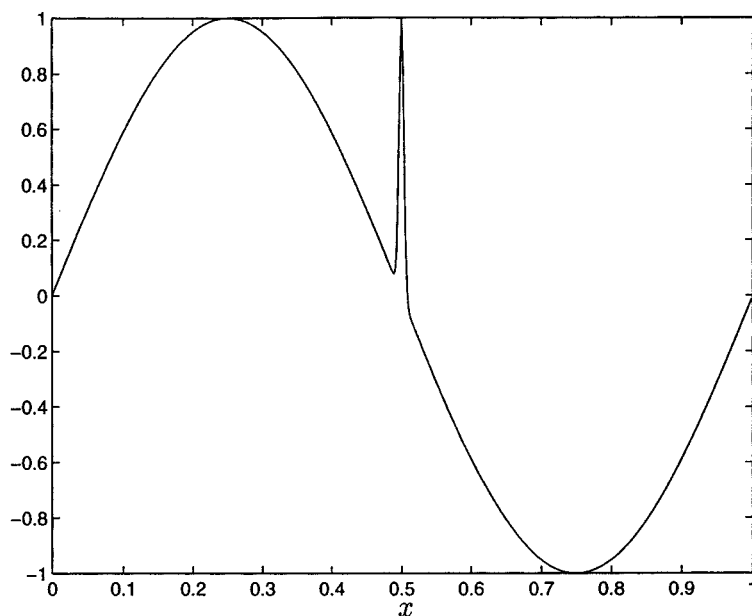


FIG. 3. Test function  $f = \sin(2\pi x) + e^{-20000(x-0.5)^2}$ .

The truncated (or sparse wavelet) representation is defined as

$$(3.4) \quad \tau_\epsilon = \mathcal{T}_\epsilon \tau = (t_\epsilon^{n-1}, \dots, t_\epsilon^{n_0}),$$

where

$$(3.5) \quad \begin{aligned} (t_\epsilon^j)(l) &= t^j(l), \quad (j, l) \in I_n^\epsilon, \\ (t_\epsilon^j)(l) &= 0, \quad (j, l) \notin I_n^\epsilon. \end{aligned}$$

We compare the error in the filter bank approximation

$$(3.6) \quad e_f = \mathcal{F}^{-k} \mathcal{T}_\epsilon \mathcal{F}^k s^n - s^n$$

with the error in the wavelet Galerkin approximation

$$(3.7) \quad e_w = r \mathcal{F}^{-k} \mathcal{T}_\epsilon \mathcal{F}^k \tilde{r} s^n - s^n.$$

We have a choice in what wavelets to use. Spline wavelets have shorter support than Daubechies wavelets with the same number of vanishing moments. Furthermore, it has been noted [28] that spline wavelets have better approximation properties than Daubechies wavelets with the same number of vanishing moments. This has also been the case for the examples in this paper. We therefore work with spline filter banks (and further on with  $\delta$  filter banks). We choose the filters  $h = \frac{\sqrt{2}}{8}(1, 3, 3, 1)$ ,  $\tilde{h} = \frac{\sqrt{2}}{64}(3, -9, -7, 45, 45, -7, -9, 3)$ , and a smooth function, with a “bump,”  $f = \sin(2\pi x) + e^{-20000(x-0.5)^2}$ , shown in Figure 3. We truncate with  $\epsilon = 10^{-3}$ . We have the finest level,  $n = 9$  with 512 points, and let  $k = 5$ . In Figure 4 we see the error when using the wavelet Galerkin method with quadratures to approximate inner products.

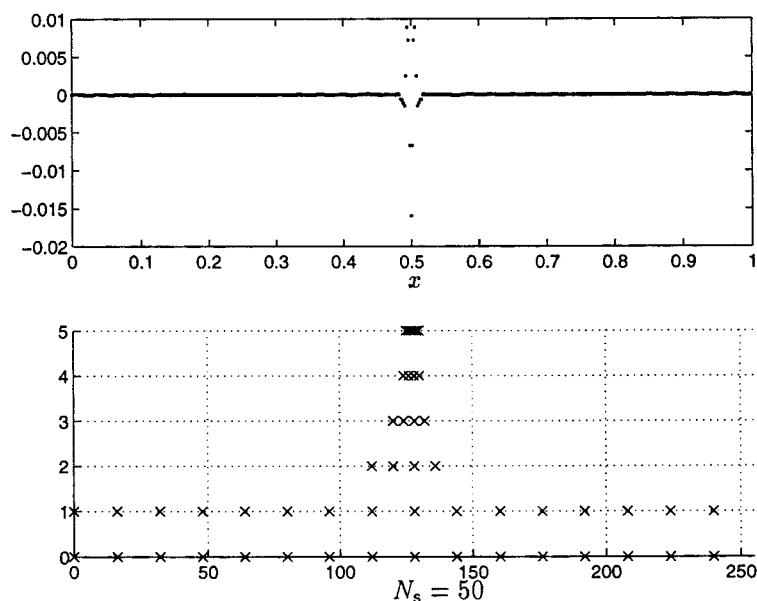


FIG. 4. Error  $e_w$  of truncated function with wavelet method,  $\epsilon = 10^{-3}$ .

We have also plotted where the retained coefficients are. In Figure 5 we see the error and coefficients when using the filter bank method. The error is about 30 times smaller (in maximum norm) with about the same number of coefficients retained. As we do not use any underlying scaling function in the filter bank method, we can also choose a dual filter that is not associated with a scaling function in  $L^2$ ,  $\tilde{h} = \frac{\sqrt{2}}{4}(-1, 3, 3, -1)$ . The result when using this filter is shown in Figure 6. We see that the error is about the same as when using the longer dual filter. We also see that only a small portion of the total number of coefficients is needed. The behavior of the  $d_k^j$  coefficients for the wavelet method is studied in [25], for example, where it is shown that they decay like  $2^{-j(s+1/2)}\|f\|_{\dot{C}^s}$  for functions in  $C^s$ . As we are working with wavelets with compact support, this means that functions that are smooth nearly everywhere ought to have sparse representations in wavelet bases. We shall prove that this is also the case when using the filter bank method.

To be more precise, we introduce the function spaces

$$(3.8) \quad \mathcal{L}^{s,m}(\mathbb{T}) = \left\{ f \in \mathbb{C}^{\mathbb{T}} : \exists N, \exists C \forall y, \exists a_1, \dots, a_N, \exists b_1, \dots, b_N : \right. \\ \left. \|f\|_{\dot{C}^s(\mathbb{T} \setminus \cup_{i=1}^N (a_i, b_i))} \leq y, \sum_{i=1}^N |b_i - a_i| \leq \frac{C}{y^m} \right\}.$$

The functions in these spaces are allowed to have a finite number of discontinuities, but the seminorms  $\|f\|_{\dot{C}^s}$  are only allowed to be large on a finite number of intervals of decreasing length, so the spaces are slightly larger than the  $C^s$  spaces. For example, the step function,  $\Theta(x - 0.5)$  belongs to  $\mathcal{L}^{s,m}$  for all  $s$  and  $m$ , and so does  $|x - 0.5|^{-\alpha}$  for all  $\alpha > 0$ ,  $m \leq 1/(s + \alpha)$  (when periodized). On the contrary, let  $f(x) = |x -$

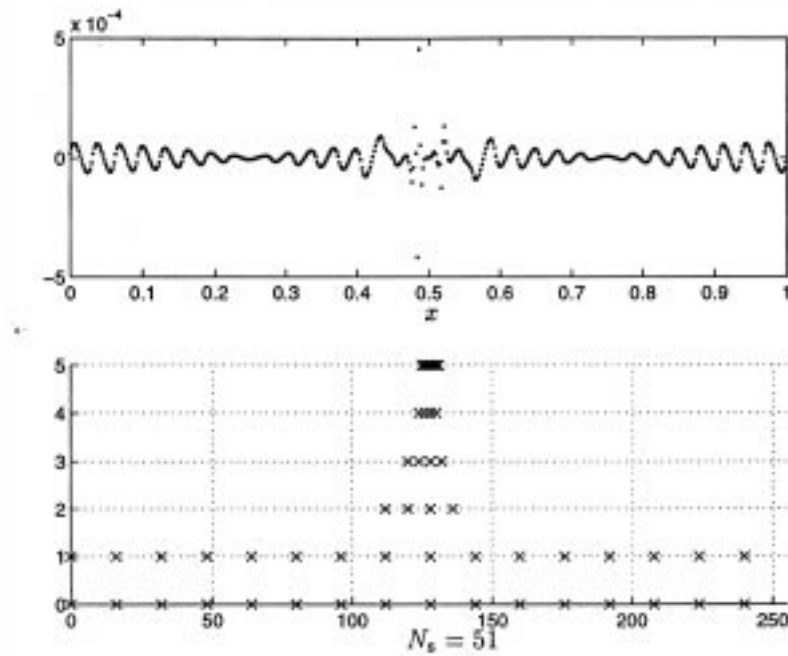


FIG. 5. Error  $e_f$  of truncated function with filter bank method,  $\epsilon = 10^{-3}$ .

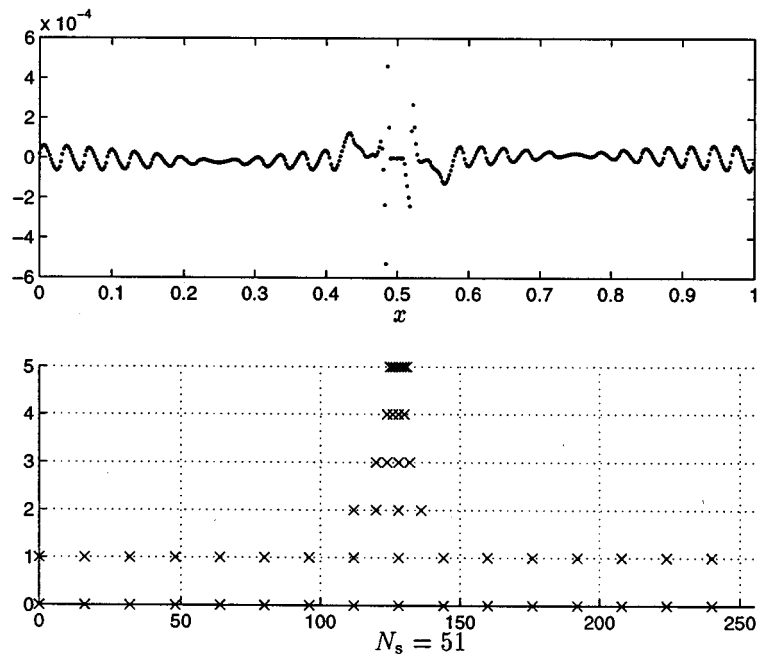


FIG. 6. Error  $e_f$  of truncated function with filter bank method and dual filter with short support,  $\epsilon = 10^{-3}$ .

$0.5|\log|x-0.5|$ ,  $x \neq 0.5$ ,  $f(0.5) = 0$ . If we enumerate the rational numbers  $q \in \mathbb{Q}$  and define  $g_s = \sum_{j=0}^{\infty} 2^{-j} f(\cdot - q_j)^s$ ,  $s \in (0, 1]$ , then  $g_s$  is in  $C^{s-\epsilon}$  for all  $\epsilon > 0$  but not in  $\mathcal{L}^{s,m}$  for any  $m$ .

We will now show that for functions in  $\mathcal{L}^{s,m}$ , the filter bank method will give a compact representation with small error. The theorems in this paper rely on the basic fact that for polynomials of a degree less than the number of vanishing moments of  $\tilde{g}$ ,  $P \in \mathcal{P}^{N-1}$ , we have  $\tilde{G}\tilde{H}^{j-1}P = 0$ , and therefore all information is kept in  $\tilde{H}^jP$ .

**THEOREM 3.1.** *Suppose  $f \in \mathcal{L}^{s,m}(\mathbb{T})$  is sampled at scale  $n$ , wavelet transformed  $k$  steps with a perfect reconstruction filter bank, where  $\tilde{g}$  has  $N > s$  zero-moments, and  $f$  is truncated with  $\mathcal{T}_\epsilon$ . Furthermore, suppose that  $\|\tilde{H}^p\|_\infty \leq C2^{\zeta_1 p}$ ,  $\|(H^*)^p\|_\infty \leq C2^{\zeta_2 p}$ . Then*

$$(3.9) \quad \|e_f\|_\infty \leq C_0(2^{\zeta_2 k} + k)\epsilon,$$

where

$$(3.10) \quad e_f = \mathcal{F}^{-k} \mathcal{T}_\epsilon \mathcal{F}^k[f]_{2^{-n}} - [f]_{2^{-n}},$$

and  $C_0$  is independent of  $n, k$ . Furthermore, the number of nonzero coefficients in  $\tau_\epsilon$ ,  $N_s$  satisfies

$$(3.11) \quad N_s \leq C_1 k + 2^{n-k} + C_2 q,$$

where

$$(3.12) \quad q = \begin{cases} 2^{\frac{\zeta_1}{s+\zeta_1}n} \epsilon^{-\frac{1}{s+\zeta_1}}, & m > \frac{1}{s+\zeta_1}, \\ k 2^{\frac{\zeta_1}{s+\zeta_1}n} \epsilon^{-\frac{1}{s+\zeta_1}}, & m = \frac{1}{s+\zeta_1}, \\ 2^{(1-ms)n} \epsilon^{-m}, & m < \frac{1}{s+\zeta_1}, \end{cases}$$

and  $C_1, C_2$  are independent of  $n, k, \epsilon$ .

*Proof.* For the error we have

$$\begin{aligned} \|e_f\|_\infty &= \left\| \sum_{j=0}^{k-1} (H^*)^j G^*(t^{n-j-1} - t_\epsilon^{n-j-1}) + (H^*)^k (t^{n-k-1} - t_\epsilon^{n-k-1}) \right\|_\infty \\ &\leq (1 + \|G^*\|_\infty) \sum_{j=0}^k \|(H^*)^j\|_\infty \|t^{n-j-1} - t_\epsilon^{n-j-1}\|_\infty \\ (3.13) \quad &\leq \sum_{j=0}^k C 2^{\zeta_2 j} \epsilon \leq C_0(2^{\zeta_2 k} + k)\epsilon. \end{aligned}$$

For the estimate of  $N_s$ , we need to know what coefficients in  $s^n$  have influence on  $(\tilde{H}^j s^n)(l)$ . We call this the *influence interval*  $\text{Infl}(\tilde{H}^j, l)$ . The coefficients that have influence on  $(\tilde{H} s^n)(l)$  are  $[2l - M(\tilde{h}), 2l - m(\tilde{h})]_{\mathbb{Z}}$ , so we get

$$\begin{aligned} \text{Infl}(\tilde{G}\tilde{H}^j, l) &= [\underline{L}, \bar{L}]_{\mathbb{Z}} \\ &= [2^{j+1}l - (2^{j+1} - 2)M(\tilde{h}) - M(\tilde{g}), 2^{j+1}l - (2^{j+1} - 2)m(\tilde{h}) - m(\tilde{g})]_{\mathbb{Z}}. \end{aligned}$$

As we are working on  $\mathbb{T}$  these coefficients should be modulo unity, but it is only the length of the interval that will be interesting so we identify the functions in  $\mathcal{L}^{s,m}(\mathbb{T})$  with 1-periodic functions, remembering that the wavelet transform will only use coefficients on one of the periods. If we suppose that  $2^{-n}[\underline{L}, \bar{L}]$  does not overlap

the  $R$  points where  $f$  may have a discontinuity, we can make a generalized Taylor expansion of  $f$  on  $2^{-n}[\underline{L}, \bar{I}]$ . We put  $y = 2^{j+1-n}l$  and get

$$(3.14) \quad \begin{aligned} f(y+x) &= f(y) + f^{(1)}(y)x + f^{(2)}(y)\frac{x^2}{2} + f^{(3)}(y)\frac{x^3}{6} + \cdots \\ &\quad + r_y(x)\frac{x^s}{s(s-1)\cdots(s-\lfloor s \rfloor + 1)}, \end{aligned}$$

where  $x \in 2^{-n}[\underline{L}, \bar{I}] - y$ , and  $\|r_y\|_\infty \leq \|f\|_{\dot{C}^s(2^{-n}[\underline{L}, \bar{I}])}$ . However, as the set of polynomials of degree  $\lfloor s \rfloor$ ,  $\mathcal{P}^{\lfloor s \rfloor}$  is closed under dilation and translation, we know that for a polynomial  $P$  of degree not higher than  $\lfloor s \rfloor$ ,  $\tilde{H}^j[P(x)]_h = [Q(x)]_{2^h}$ , where  $Q$  is also a polynomial of degree  $\lfloor s \rfloor$ . Therefore, the polynomial part of (3.14)  $P$  will be mapped to zero  $(\tilde{G}\tilde{H}^j[P]_{2^{-n}})(l) = 0$ , as the moment conditions on  $\tilde{g}$  make it map polynomials of degree  $N-1$  or less to zero. For the last term in (3.14) we have

$$(3.15) \quad \begin{aligned} |d_l^{n-j-1}| &= |(\tilde{G}\tilde{H}^j[f]_{2^{-n}})(l)| = \left| \left( \tilde{G}\tilde{H}^j \left[ r_y(x) \frac{x^s}{s(s-1)\cdots(s-\lfloor s \rfloor + 1)} \right]_{2^{-n}} \right)(l) \right| \\ &\leq \frac{1}{s(s-1)\cdots(s-\lfloor s \rfloor + 1)} \|\tilde{G}\|_\infty \|\tilde{H}^j\|_\infty \|r_y(x)\chi_{2^{-n}[\underline{L}, \bar{I}]} \|x^s\chi_{2^{-n}[\underline{L}, \bar{I}]} \|_\infty \\ &\leq c_0 2^{(j-n)s} \|\tilde{H}^j\|_\infty \|f\|_{\dot{C}^s(2^{-n}[\underline{L}, \bar{I}])} \leq c_1 2^{(j-n)s+\zeta_1 j} \|f\|_{\dot{C}^s(2^{-n}[\underline{L}, \bar{I}])}, \end{aligned}$$

where we have used  $2^{-n}(\bar{I} - \underline{L}) \leq c_2 2^{j-n}$ , and  $\|r_y\|_\infty \leq \|f\|_{\dot{C}^s(2^{-n}[\underline{L}, \bar{I}])}$ .

We have two bounds on the number of coefficients on each level  $N_s^{n-j}$  that have  $|d_k^{n-j}| \geq \epsilon$ . The first is trivial,

$$(3.16) \quad N_s^{n-j} \leq 2^{n-j}.$$

For the second bound, we use (3.15) to see that

$$|d_l^{n-j-1}| \geq \epsilon \Rightarrow \|f\|_{\dot{C}^s(2^{-n}[\underline{L}, \bar{I}])} \geq c_1^{-1} \epsilon 2^{(n-j)s-\zeta_1 j}.$$

However, from the definition of  $\mathcal{L}^{s,m}$ , the regions where this can be satisfied are at most  $R$  intervals of total length  $c_3 \epsilon^{-m} 2^{-m(n-j)s+\zeta_1 mj}$ , so the total number of coefficients that have an influence set that overlaps these regions is at most

$$(3.17) \quad N_s^{n-j-1} \leq \sum_{i=1}^R \frac{c_4 2^{1+j-n} + (b_i - a_i)}{2^{1+j-n}} = Rc_4 + c_3 2^{(n-j)(1-ms)+\zeta_1 mj-1} \epsilon^{-m}.$$

We first assume that  $m > \frac{1}{s+\zeta_1}$  so that the exponent in (3.17) is less than zero when summing over  $j$ . The break-even between the two bounds lies near  $j = \frac{\log_2 \epsilon + ns}{s+\zeta_1}$ , and we divide the sum into two parts, where only the largest terms are interesting as both sums form decreasing geometric series. The total number of coefficients therefore satisfies

$$(3.18) \quad \begin{aligned} \sum_{j=1}^k N_s^{n-j} &\leq Rc_4 k + \frac{2^n}{2} \sum_{j=0}^{k-1} 2^{-j} \min\{1, c_3 2^{-ms(n-j)+\zeta_1 mj} \epsilon^{-m}\} \\ &\leq Rc_4 k + c_5 2^n \epsilon^{-m} 2^{-msn+(ms+m\zeta_1-1)(\log_2 \epsilon + ns)/(s+\zeta_1)} \\ &\quad + c_6 2^n 2^{-(\log_2 \epsilon + ns)/(s+\zeta_1)} = C_1 k + C_2 2^{\frac{\zeta_1}{s+\zeta_1} n} \epsilon^{-\frac{1}{s+\zeta_1}}. \end{aligned}$$

We now assume that  $m \leq \frac{1}{s+\zeta_1}$ . In this case both (3.16) and (3.17) are increasing when  $j$  decreases. Therefore, there is no gain in summing over both series, and we estimate the sum with

$$\sum_{j=0}^{k-1} Rc_4 + c_3 2^{(n-j)(1-ms)+\zeta_1 mj-1} \epsilon^{-m},$$

which for  $m < \frac{1}{s+\zeta_1}$  is less than or equal to  $C_1 k + C_2 2^{(1-ms)n} \epsilon^{-m}$  and for  $m = \frac{1}{s+\zeta_1}$  is less than or equal to  $C_1 k + C_2 k 2^{(1-ms)n} \epsilon^{-m}$ .

Finally, the number of coefficients in  $s^{n-k}$  is bounded by  $2^{n-k}$ , leading to the total bound on the coefficients.  $\square$

We see here that it is not important for  $g$  to have vanishing moments. The only thing important for the dual filter  $\tilde{h}$  is to satisfy the perfect reconstruction identity (2.37) and for  $\zeta_1$  to be small. The following theorem gives bounds on  $\zeta_1$ .

**THEOREM 3.2.** *Let  $(h, g, \tilde{h}, \tilde{g})$  be a perfect reconstruction filter bank and  $\tilde{L} = \text{length } \tilde{h}$ . Define*

$$(3.19) \quad \tilde{M} = \frac{1}{2} \tilde{h} * \tilde{h} = (\tilde{m}_{-\tilde{L}+1}, \dots, \tilde{m}_0, \dots, \tilde{m}_{\tilde{L}-1}),$$

and the  $(2\tilde{L}-1) \times (2\tilde{L}-1)$  matrix

$$(\tilde{\mathbf{P}})_{j,k} = \begin{cases} 2\tilde{m}_{2j-k-\tilde{L}}, & |2j-k-\tilde{L}| < \tilde{L}, \\ 0, & |2j-k-\tilde{L}| \geq \tilde{L}. \end{cases}$$

Then

$$(3.20) \quad \|\tilde{H}^p\|_\infty \leq C(\rho(\tilde{\mathbf{P}}) + \epsilon)^{p/2},$$

where  $\rho(\tilde{\mathbf{P}})$  is the spectral radius of  $\tilde{\mathbf{P}}$ ,  $\epsilon > 0$  is arbitrary (and we can choose  $\epsilon = 0$  if the eigenvalue with the largest absolute value is nondegenerate) and  $C$  is independent of  $p$ .

*Proof.* By  $H|_{a,b}$  we mean the restriction of  $H$  to  $[a, b]_{\mathbb{Z}}$ ,  $H|_{a,b} u \mapsto ((Hu)(a), \dots, (Hu)(b))$ . We have the following norm relations:

$$\|\tilde{H}^p\|_\infty = \|\tilde{H}^p|_{0,0}\|_\infty \leq C2^{p/2} \|\tilde{H}^p|_{0,0}\|_2 \leq C2^{p/2} \|\tilde{H}^p\|_2.$$

However,

$$\widehat{\tilde{H}^p f} = \frac{1}{\sqrt{2}} \widehat{\downarrow \tilde{h} f} = \frac{1}{2\sqrt{2}} \left( \hat{h}(-\omega/2) \hat{f}(\omega/2) + \hat{h}(-\omega/2 + \pi) \hat{f}(\omega/2 + \pi) \right),$$

leading to

$$\widehat{\tilde{H}^p f} = \frac{1}{2^p} \sum_{k=0}^{2^p-1} \left( \prod_{j=1}^p \frac{1}{\sqrt{2}} \hat{h}(2^{-j}(-\omega + 2\pi k)) \right) \hat{f}(2^{-p}(\omega + 2\pi k)).$$

Parseval's identity gives us

$$\|\tilde{H}^p\|_2 = \sup_{\hat{f} \in L^2(0, 2\pi)} \frac{\left\| \frac{1}{2^p} \sum_{k=0}^{2^p-1} \left( \prod_{j=1}^p \frac{1}{\sqrt{2}} \hat{h}(2^{-j}(-\omega + 2\pi k)) \right) \hat{f}(2^{-p}(\omega + 2\pi k)) \right\|_2}{\|\hat{f}(\omega)\|_2},$$



and by using  $|\hat{h}(\omega)|^2/2 = \hat{M}(\omega)$  and the Cauchy–Schwarz and Hölder inequalities we get

$$(3.21) \quad \|\tilde{H}^p\|_2^2 \leq \frac{1}{2^p} \sup_{\omega \in [0, 2\pi)} \sum_{k=0}^{2^p-1} \prod_{j=1}^p \hat{M}(2^{-j}(-\omega + 2\pi k)).$$

If we define the operator

$$(3.22) \quad (P_{\hat{M}} f)(\xi) = \frac{1}{2}(\hat{M}(-\xi/2)f(\xi/2) + \hat{M}(-\xi/2 + \pi)f(\xi/2 + \pi)),$$

acting on  $2\pi$ -periodic functions, (3.21) can be written

$$\|\tilde{H}^p\|_2^2 \leq \|P_{\hat{M}}^p 1\|_\infty.$$

For trigonometric polynomials of the form  $Q(\omega) = \sum_{k=-\tilde{L}+1}^{\tilde{L}-1} a_k e^{ik\xi}$ , we have that  $(P_{\hat{M}} Q)(\omega) = \sum_{k=-\tilde{L}+1}^{\tilde{L}-1} b_k e^{ik\xi}$ , where  $\mathbf{b} = \tilde{\mathbf{P}}\mathbf{a}/2$ , and  $(\mathbf{a})_j = a_{j-\tilde{L}}$ ,  $(\mathbf{b})_j = b_{j-\tilde{L}}$ , so

$$\|P_{\hat{M}}^p 1\|_\infty \leq c \|P_{\hat{M}}^p 1\|_2 \leq c_2 \frac{(\rho(\tilde{\mathbf{P}}) + \epsilon)^p}{2^p},$$

altogether leading to

$$\|H^p\|_\infty \leq C(\rho(\tilde{\mathbf{P}}) + \epsilon)^{p/2}. \quad \square$$

We remark that a necessary condition on  $\tilde{\mathbf{P}}$  for  $\tilde{\varphi}$  to generate a Riesz basis of  $L^2$  is that  $\rho(\tilde{\mathbf{P}}) = 1$  (see, e.g., [7]), so for such choices of  $\tilde{h}$  we have  $\zeta_1 = 0$ . For bounds on  $\|(H^*)^p\|_\infty$  we have the following result.

**THEOREM 3.3.** *Let  $(h, g, \tilde{h}, \tilde{g})$  be a perfect reconstruction filter bank, such that there is a  $\varphi$  (with compact support) in an MRA associated with  $h$  (but not necessarily with  $\tilde{h}$ ), and  $\varphi \in L^\infty$ . Then*

$$(3.23) \quad \|(H^*)^p\|_\infty \leq C,$$

where  $C$  is independent of  $p$ .

*Proof.* As  $\varphi$  generates a Riesz basis of  $L^2$  we know from [25] that by defining the dual basis function

$$\hat{\varphi}(\omega) = \frac{\hat{\varphi}(\omega)}{\sum_k |\hat{\varphi}(\omega + 2\pi k)|^2},$$

we get  $\langle 2^{p/2}\varphi(2^p \cdot -k), 2^{p/2}\tilde{\varphi}(2^p \cdot) \rangle = \delta_k$ , and  $\tilde{\varphi} \in L^1$ . For an arbitrary  $c$ , we wish to estimate  $\|(H^*)^p c\|_\infty$ , so we define  $f = \sum_k c_k \varphi_{0,k}$ . From Hölder's inequality we get

$$|\langle f, 2^{p/2}\tilde{\varphi}(2^p \cdot -m) \rangle| \leq \|f\|_\infty \|2^{p/2}\tilde{\varphi}(2^p \cdot -m)\|_1 \leq 2^{-p/2} \|\varphi\|_\infty \|\tilde{\varphi}\|_1 \|c\|_\infty (\text{length } \tilde{h} - 1).$$

However, from (2.19), (2.43) we have  $\langle f, 2^{p/2}\tilde{\varphi}(2^p \cdot -m) \rangle = 2^{-p/2}((H^*)^p c)(m)$ , so

$$\|(H^*)^p c\|_\infty \leq (\text{length } \tilde{h} - 1) \|\varphi\|_\infty \|\tilde{\varphi}\|_1 \|c\|_\infty. \quad \square$$

As we typically will require many vanishing moments for  $\tilde{g}$  to get a high approximation order, we usually will have  $\varphi \in L^2 \cap L^\infty$ , so the conditions of Theorem 3.3 will be satisfied. The strength of the filter bank method lies in choosing  $\tilde{h}$  not corresponding to any  $\varphi$ .

TABLE 3.1  
Bounds for  $\beta$  (spline), filter banks.

Name	$\zeta_1$	$\frac{\zeta_1}{\zeta_1+N}$	$\frac{1}{\zeta_1+N}$
$\beta_0$	0	0	1
$\beta_1$	0.16096	0.07448	0.46275
$\beta_2$	0.55923	0.15712	0.28095
$\beta_3$	1.28630	0.24332	0.18916
$\beta_4$	1.82486	0.26738	0.14652
$\beta_5$	2.59075	0.30157	0.11640

There are of course other bounds. From Young's inequality we get

$$\|\tilde{H}^p\|_\infty \leq \left( \frac{\|\tilde{h}\|_1}{\sqrt{2}} \right)^p,$$

$$\|(H^*)^p\|_\infty \leq (\sqrt{2}\|h\|_1)^p$$

(where, on the right-hand side, we interpret  $h, \tilde{h}$  as filters, *not* as operators), but the bound in Theorem 3.2 seems to be close to optimal. The filter banks we have listed in the tables have  $h$  corresponding to  $\varphi \in L^2 \cap L^\infty$ , and thus  $\zeta_2 = 0$ . The filter banks in Tables 2.1 and 2.3 also have  $\tilde{\varphi} \in L^2$ , so for these  $\zeta_1 = 0$ . The filter banks in Tables 2.2 and 2.4 do *not* have  $\tilde{h}$  corresponding to  $\tilde{\varphi} \in L^2$ . However, the bounds obtained from Young's inequality immediately imply that  $\zeta_1 = 0$  for the  $\delta$ -filters. In Table 3.1 we list the bounds found in Theorem 3.2 for the spline wavelets and the exponents involved in Theorem 3.1 (where  $N$  denotes the number of vanishing moments for  $\tilde{g}$ , which will govern the decay for  $\mathcal{L}^{s,m}$  when  $s > N$ ). We choose the bound for  $m > \frac{1}{s+\zeta_1}$ , as this will be "right" when  $f$  has no worse singularities than discontinuities.

For the filter  $\tilde{h} = \frac{\sqrt{2}}{4}(-1, 3, 3, -1)$ , we get  $\zeta_1 \leq 0.559$  and, as  $h = \frac{\sqrt{2}}{4}(1, 3, 3, 1)$  gives a  $\tilde{g}$  with three vanishing moments, the  $\beta_2$  filter bank can be used without problems in numerical calculations. We also note that from [7], [14] it follows that for  $\zeta_1 > 0$  we have

$$(3.24) \quad \tilde{\varphi} \in H^{-\zeta_1-\epsilon}, \quad \epsilon > 0,$$

so  $\zeta_1$  gives a measure of how "far" from  $L^2$  the corresponding wavelet (now defined in a distributional sense) is.

It might also be enlightening to check the condition numbers of the involved transforms, i.e., to check how  $\kappa(\mathcal{F}^n) \stackrel{\text{def}}{=} \|\mathcal{F}^n\|_\infty \|(\mathcal{F}^n)^{-1}\|_\infty$  depends on the order of the filter banks and of  $n$ . In Figure 7 we show the condition number as a function of  $n$  for the spline and  $\delta$ -filter banks. We see that  $\kappa$  grows slowly with  $n$  for the  $\delta$ -filter bank and for the low-order spline filter banks. For the higher order splines the growth is fast, which seems to imply problems with accuracy. However, as shown in Theorem 3.1 it is only  $\|(\mathcal{F}^n)^{-1}\|_\infty$  that influences the accuracy, whereas  $\|\mathcal{F}^n\|_\infty$  influences the number of significant coefficients in the truncation step. This rule is also valid with finite precision calculations as long as  $\epsilon$  in the truncation is much larger than the machine precision, which is the case in the examples of this paper. In Figure 8 we show how  $\|(\mathcal{F}^n)^{-1}\|_\infty$  grows as a function of  $n$  for the spline filter banks. We see that the growth is slow, so the loss in accuracy is kept under control (although the fast growth of  $\|\mathcal{F}^n\|_\infty$  might result in a larger number of significant coefficients in the truncation step).

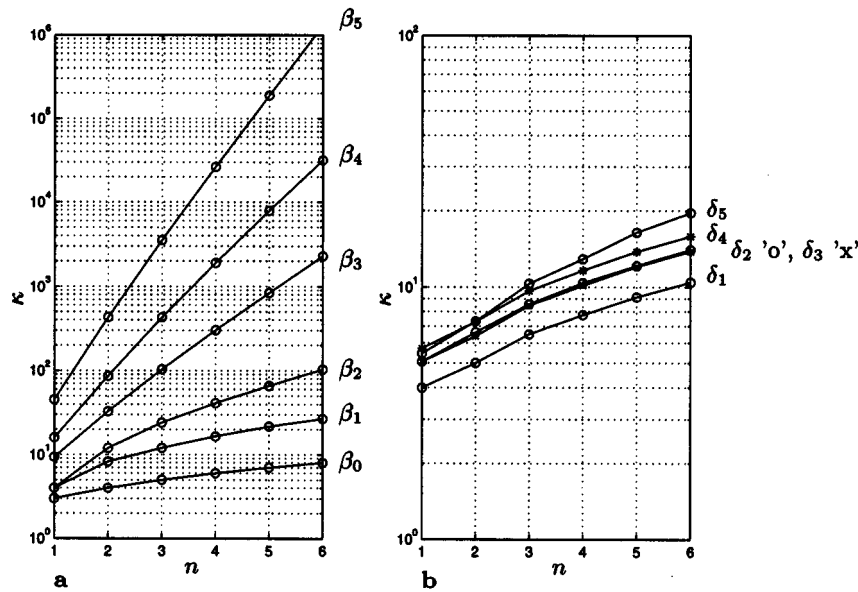


FIG. 7. Condition number  $\kappa(\mathcal{F}^n)$  as a function of  $n$  (a) for spline filters and (b) for  $\delta$ -filters.

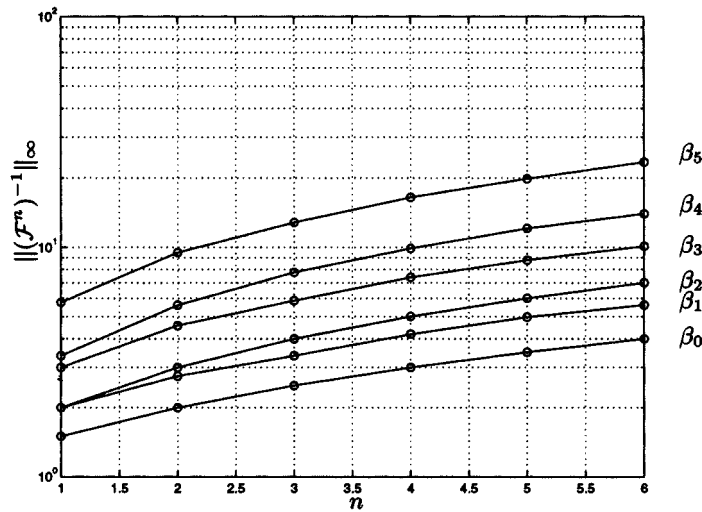


FIG. 8. Norm  $\|(\mathcal{F}^n)^{-1}\|_\infty$  as a function of  $n$  for spline filters.

Another consequence of Theorem 3.1 is that we do not have to worry about the Gibbs phenomena when using the filter bank method as in the wavelet Galerkin case. As an example we take the function in Figure 9  $f(x) = \sin(2\pi x) - \Theta(x - 1/2) + 1/2$ , transform it with the wavelet Galerkin method, and compare this with the result when using the filter bank method. The result is shown in Figure 10. With the wavelet Galerkin method, the error will not decrease in max-norm as  $\epsilon$  decreases or  $n$  increases. In contrast, when using the filter bank method the error decreases when

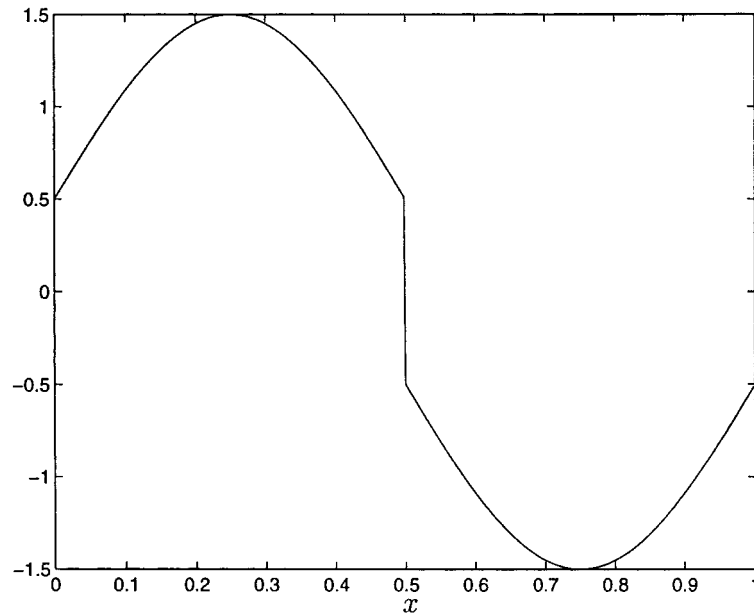


FIG. 9. Test function  $f(x) = \sin(2\pi x) - \Theta(x - \frac{1}{2}) + \frac{1}{2}$ .

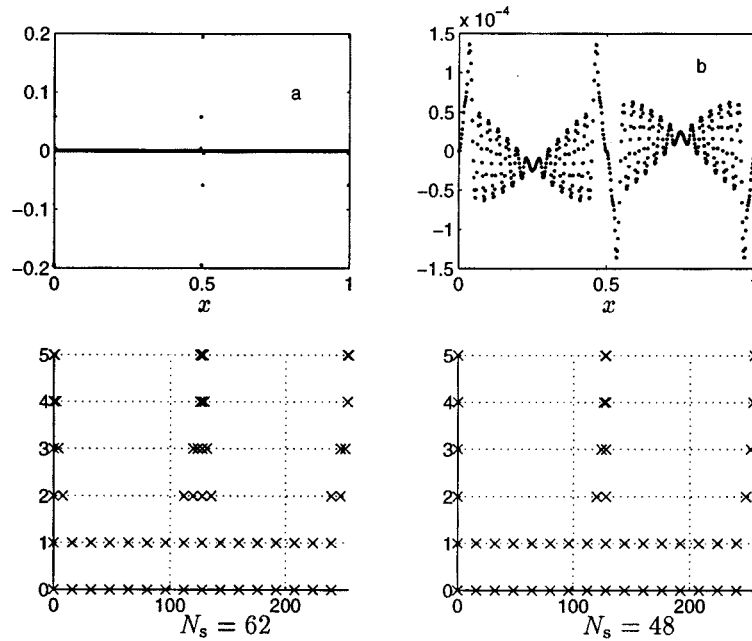


FIG. 10. Error of truncated function with (a) wavelet method and (b) filter bank method,  $\epsilon = 10^{-3}$ .

$\epsilon$  decreases in accordance with Theorem 3.1.

We also get good results with a “nasty” function like  $f(x) = \sin(2\pi x) + \frac{1}{1000(x-0.5)^2}$ ,  $x \neq 0.5$ ,  $f(0.5) = 0$ , the absolute value of which is plotted in  $\log_{10}$  scale in Figure 11.

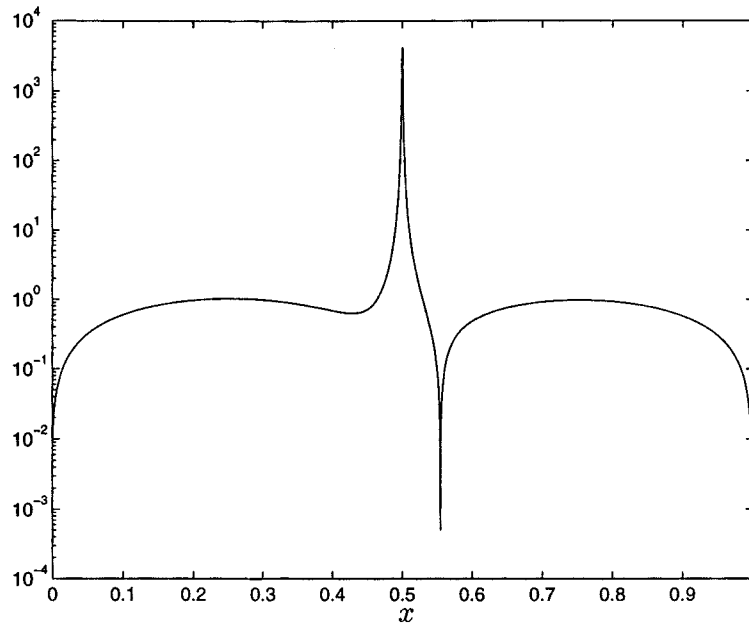


FIG. 11. Test function  $f(x) = \sin(2\pi x) + \frac{1}{1000(x-0.5)^2}$  absolute value plotted in  $\log_{10}$  scale.

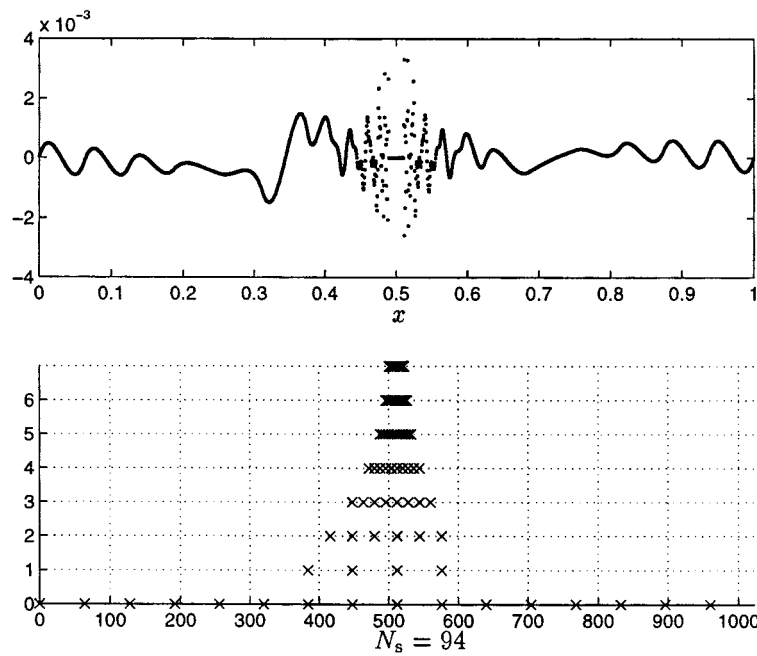


FIG. 12. Error of truncated function with filter bank method,  $\epsilon = 10^{-3}$ .

We choose  $n = 11$ ,  $k = 7$ ,  $\epsilon = 5 \times 10^{-4}$ . The error is shown in Figure 12. We see that it is small,  $\|e_f\|_\infty = 3.3 \times 10^{-3}$ , with a compression factor of  $2048/94 = 21.8$ .

All the calculations in this and subsequent sections were computed in double precision. The error estimate in Theorem 3.1 gives us a hint that the maximum norm is the “right” norm for the error. *We will therefore measure all errors in the maximum norm.*

**3.2. Differentiation of truncated functions.** We are now going to introduce a way to differentiate a truncated representation,  $\tau_\epsilon$ . As we have point values, it seems logical to use a finite difference approximation. In the language of filters, a finite difference operator is simply a convolution operator (where the coefficients depend on the grid size  $h$  used in the sampling), with a finite number of nonzero coefficients.

**DEFINITION 3.4.** *A convolution operator, represented by a filter  $p(h)$ , is said to be an  $N$ th-order approximation of  $\frac{d^s}{dx^s}$  if  $p(h)[x^k]_h = [\frac{d^s(x^k)}{dx^s}]_h$ ,  $k = 0, \dots, N + s - 1$ .*

An  $N$ th-order approximation of  $\frac{d^s}{dx^s}$  will be of the form  $p(h) = \frac{1}{h^s}(q_m, \dots, q_M)$  and the order conditions that will be satisfied are

$$(3.25) \quad \frac{1}{s!} \sum_{j=m}^M q_{-j} j^k = \delta_{k-s}, \quad k = 0, \dots, N + s - 1.$$

When differentiating we could, of course, transform the truncated representation back to the finest level and apply the finite difference operator to this representation, but this would not make any use of the sparse representation we have. Instead we would like to apply the operator on a coarse scale where this can be used. The following theorem justifies such an approach.

**THEOREM 3.5.** *Suppose  $d(h)$  is an  $N$ th-order finite difference approximation of  $\frac{d^s}{dx^s}$ . If  $(h, g, \tilde{h}, \tilde{g})$  is a perfect reconstruction filter bank, and  $\tilde{g}$  has  $N_2$  zero-moments, then*

$$(3.26) \quad (H^*)^j d(2^j h) \tilde{H}^j [x^k]_h = \left[ \frac{d^s(x^k)}{dx^s} \right]_h, \quad k = 0, \dots, \min\{N, N_2\} + s - 1.$$

*Proof.* We know that for any  $P \in \mathcal{P}^k$ ,  $\tilde{H}[P(x)]_h = [Q(x)]_{2h}$ , where  $Q \in \mathcal{P}^k$ . The theorem now follows from

$$(3.27) \quad \tilde{H} \left[ \frac{d^s P}{dx^s} \right]_h = \left[ \frac{d^s Q}{dx^s} \right]_{2h}.$$

This is seen for  $j = 1$ , as  $d(2h)$  differentiates every polynomial in  $\mathcal{P}^{N+s-1}$  exactly, and every polynomial in  $\mathcal{P}^{N_2-1}$  is mapped to 0 by  $\tilde{G}$ , so

$$\left[ \frac{d^s P}{dx^s} \right]_h = (H^* \tilde{H} + G^* \tilde{G}) \left[ \frac{d^s P}{dx^s} \right]_h = H^* \tilde{H} \left[ \frac{d^s P}{dx^s} \right]_h = H^* \left[ \frac{d^s Q}{dx^s} \right]_{2h} = H^* d(2h)[Q]_{2h},$$

$$P \in \mathcal{P}^{\min\{N, N_2\} + s - 1}.$$

By induction this is true for all  $j$ . It remains to show (3.27) which is a consequence of differentiation commuting with convolution

$$\left[ \frac{d^s Q}{dx^s} \right]_{2h} = \left[ \frac{d^s}{dx^s} \frac{1}{\sqrt{2}} \tilde{h} P \right]_{2h} = \left[ \frac{1}{\sqrt{2}} \downarrow \tilde{h} \frac{d^s}{dx^s} P \right]_h = \tilde{H} \left[ \frac{d^s P}{dx^s} \right]_h,$$

where in the intermediate steps we used semicontinuous convolution and continuous downsampling.  $\square$

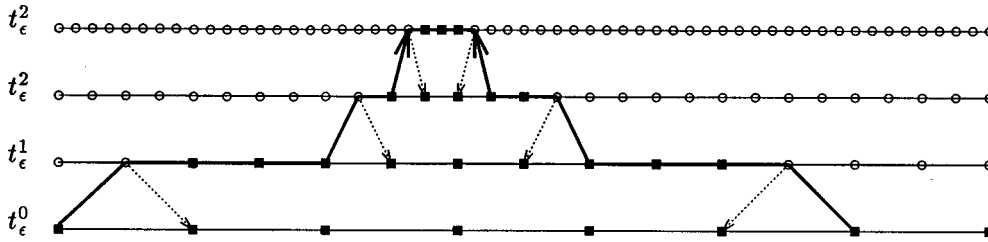


FIG. 13.  $\Lambda$ -cycle for fast differentiation of  $\epsilon$ -truncated wavelet expansion  $\tau_\epsilon$ .

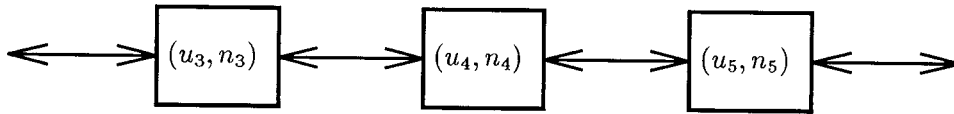


FIG. 14. Linked list representation of sparse grid functions.

This theorem leads us to a fast algorithm for differentiating an  $\epsilon$ -truncated wavelet representation of a function. We simply apply the finite difference operator locally on each scale  $j$ , where there are no  $t_\epsilon$  coefficients on finer scales. As the inverse wavelet transform is local, we can keep the number of operations at  $\mathcal{O}(N_s)$ . Convolution leads to an increase in support, so we need to add some extra coefficients at the edge points, so that the *final* coefficients are right. Furthermore, the differentiated function might have significant coefficients in new regions. We should therefore add a region where new coefficients might appear. The simplest way to do this is to study some more coefficients at the “edges” of each  $t_\epsilon^j$ . The total algorithm will then go from the coarsest scale, up to the finest scale, and down to the coarsest, as shown in Figure 13. As this cycle is opposite to the so-called *V*-cycle in a multigrid algorithm (which goes from fine scales to coarse), we call this a  $\Lambda$ -cycle.

The addition of coefficients at “edges” of  $t_\epsilon^j$  is done through the operation **smear**. Until now we have viewed the truncated transformed signal as grid functions where many of the coefficients are zero. As we need to keep track of where the nonzero coefficients are to make the algorithm efficient and we do not want to waste memory on coefficients that are zero, we introduce a different, *linked list* representation that is efficient for a grid function with many zero coefficients.

**DEFINITION 3.6.** A sparse grid function is a (possibly bi-infinite) ordered list of pairs of numbers

$$u = (\dots, (u_{k-1}, n_{k-1}), (u_k, n_k), (u_{k+1}, n_{k+1}), \dots) \quad \forall j : u_j \in \mathbb{C}, n_j \in \mathbb{Z}$$

such that  $n_{j+1} > n_j \forall j$ .

We also define the *used set*  $S_u = \{k : \exists y : (y, k) \in u\}$ . The interpretation of this definition is that  $u(k) = y$  if  $(y, k) \in u$ , and  $u(k) = 0$  if  $k \notin S_u$ . A grid function is thus represented as in Figure 14. This sparse representation is used for the various numerical examples in the subsequent sections. The adding of  $t$  extra coefficients at “edges” of a sparse grid function can now formally be defined as follows.

**DEFINITION 3.7.** The sparse grid function  $v = \mathbf{smear}_t(u)$  has its coefficients defined by  $(x, s) \in v \Leftrightarrow (x, s) \in u$ , or  $(x = 0, k \notin S_u, \exists r, r \in S_u, |r - s| \leq t)$ .

We also define an approximate left inverse of the smearing, **sharpen**, that takes





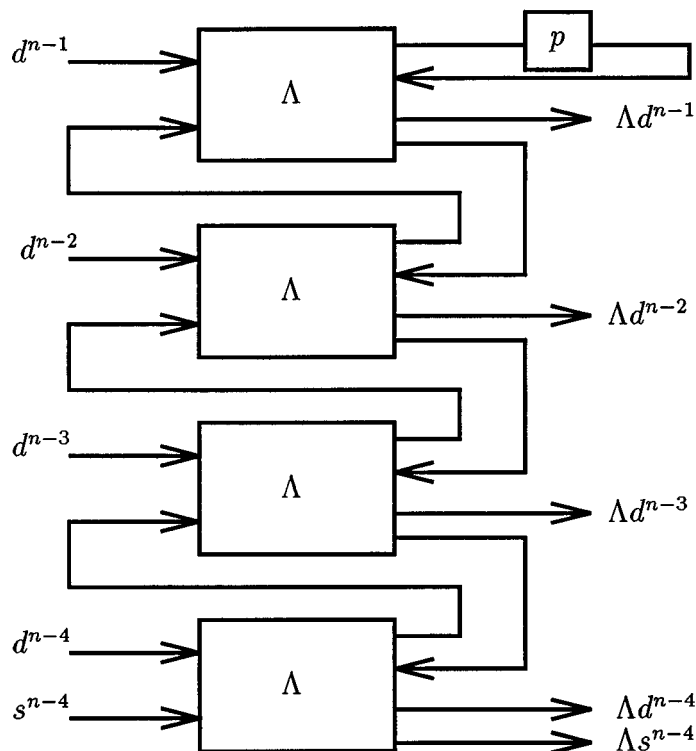


FIG. 16. Total sparse differentiation with  $\Lambda$ -cycle.

$\Lambda$ -cycle is

$$(3.28) \quad \text{OP}_\Lambda = 2 + 2ac + 2bc + c^2 - ab + (6c + 2b)s + (2a - 1)N,$$

compared with applying  $p$  directly on the finer level

$$(3.29) \quad \text{OP}_{\text{full}} = (4a - 2)N.$$

We here assume that all coefficients within the length of the filters are nonzero. The  $\beta_2$  filter bank has  $b = c = 4$  and with the standard fourth-order difference operator with  $a = 5$  we get  $\text{OP}_\Lambda = 70 + 32s + 7N$ , which should be compared with the work done by applying the operator directly on the finest grid  $14N$ , so the break-even is when  $70 + 32s = 7N$ . This gives us an idea of what the potential gains of the sparse method is. For the  $\delta_3$ -filter, the operational count will be  $-8 + 20s + 7N$ .

We now study the same test function  $f = \sin(2\pi x) + e^{-20000(x-0.5)^2}$  with the same filter bank  $\beta_2$ , the centered fourth-order finite difference approximation of  $\frac{d}{dx}$ ,  $d(h) = \frac{1}{h}(-\frac{1}{12}, \frac{2}{3}, 0, -\frac{2}{3}, \frac{1}{12})$  and we truncate with  $\epsilon = 10^{-4}$ . We choose the finest level  $n = 10$  and transform  $k = 4$  levels forward.

In Figure 17 the derivative of this function is shown, and in Figure 18 we see the error in the approximated derivative when approximating with three different methods. The first method is simply using the finite difference filter on the function

$$(3.30) \quad e_u = d(2^{-n})[f]_{2^{-n}} - \left[ \frac{df}{dx} \right]_{2^{-n}}.$$

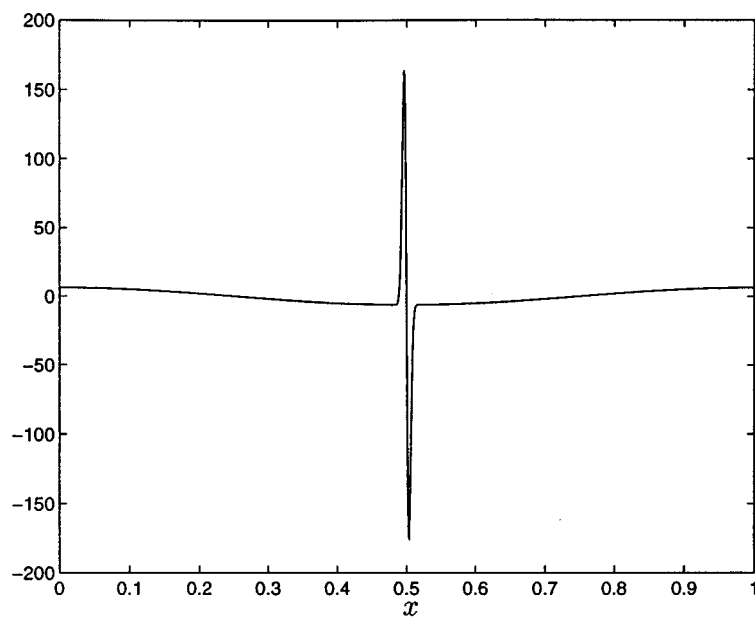


FIG. 17. Derivative of  $f = \sin(2\pi x) + e^{-20000(x-0.5)^2}$ .

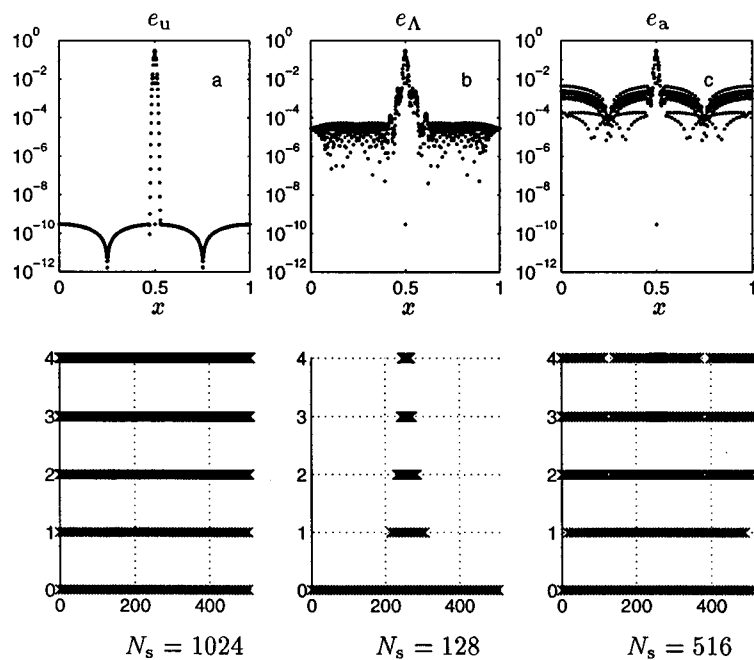


FIG. 18. Error and sparse representation for three methods to differentiate (a) untruncated, (b)  $\Lambda$ -cycle, and (c) full inverse transform.

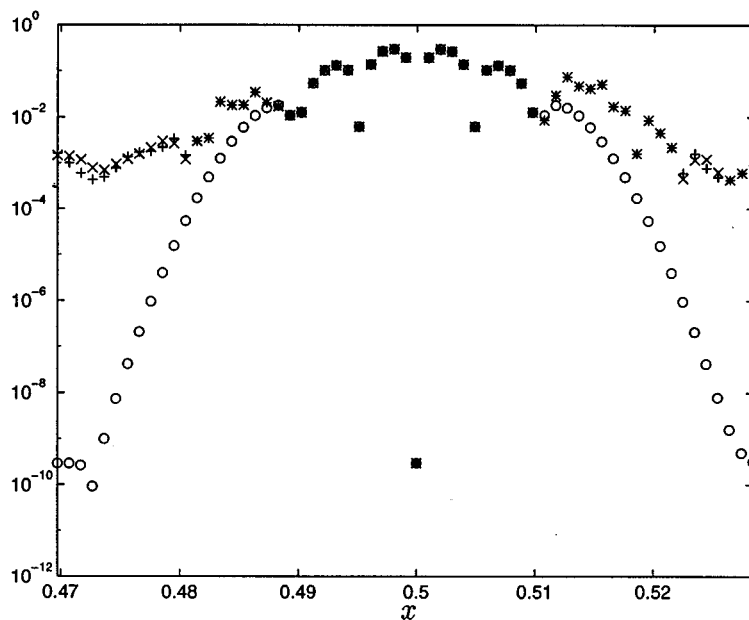


FIG. 19. Error around “spike” for untruncated “o,”  $\Lambda$ -cycle “+,” full inverse transform “x.”

The second method is using the  $\Lambda$ -cycle and truncating the result

$$(3.31) \quad e_{\Lambda} = \mathcal{F}^{-k} \mathcal{T}_{\epsilon} \Lambda_d \mathcal{T}_{\epsilon} \mathcal{F}^k [f]_{2^{-n}} - \left[ \frac{df}{dx} \right]_{2^{-n}}.$$

The last method is transforming the truncated signal all the way back to the finest level everywhere, performing  $d(2^{-n})$  on this signal, transforming forward, and truncating

$$(3.32) \quad e_a = \mathcal{F}^{-k} \mathcal{T}_{\epsilon} \mathcal{F}^k d(2^{-n}) \mathcal{F}^{-k} \mathcal{T}_{\epsilon} \mathcal{F}^k [f]_{2^{-n}} - \left[ \frac{df}{dx} \right]_{2^{-n}}.$$

The error in Figure 18 is plotted in  $\log_{10}$  scale, because of the large ranges. The lower part of the figure shows the nonzero coefficients of the representation. We see that the untruncated approximation performs best, but the  $\Lambda$ -cycle performs better than the approach of transforming back to the finest level everywhere, which seems to introduce high-frequency errors, which ruins the sparseness of the representation. In max-norm the  $\Lambda$ -cycle performs as well as the untruncated approximation as seen in Figure 19, where the error for the three methods is shown around the “spike.” This is natural, as the wavelet method detects local high frequencies and the  $\Lambda$ -cycle transforms all the way back in such regions.

The index  $r$  in  $\mathbf{smear}_{t+r}$  decides how many extra coefficients at the edges of the sparse grid function that might be added. Unfortunately, extra coefficients might be needed at a *higher level* in the representation. This will, e.g., be the case when solving the Burgers equation, where a shock forms as time increases. The  $\Lambda$ -cycle described will then be inappropriate, and we need to do some extra work to find a good approximation of the derivative. The trick is to transform back an extra level *everywhere* to see if the frequencies increase in some region. This leads to a



Downloaded 12/29/12 to 129.173.72.87. Redistribution subject to SIAM license or copyright; see <http://www.siam.org/journals/ojsa.php>



Downloaded 12/29/12 to 129.173.72.87. Redistribution subject to SIAM license or copyright; see <http://www.siam.org/journals/ojsa.php>

Downloaded 12/29/12 to 129.173.72.87. Redistribution subject to SIAM license or copyright; see <http://www.siam.org/journals/ojsa.php>

Downloaded 12/29/12 to 129.173.72.87. Redistribution subject to SIAM license or copyright; see <http://www.siam.org/journals/ojsa.php>

**3.3. Multiplication of truncated functions.** The success of a multiplication  $\Lambda$ -cycle depends, just as in the case of differentiation, on whether polynomials can be treated on coarse scales. We need to determine when

$$(H^*)^j(\tilde{H}^j[x^k]_h \times \tilde{H}^j[x^s]_h) = [x^{k+s}]_h,$$

which is needed for multiplication of two sparse wavelet representations, and when

$$(H^*)^j([x^k]_{2^j h} \times \tilde{H}^j[x^s]_h) = [x^{k+s}]_h,$$

which is needed for direct multiplication of a sparse wavelet representation and a function. Here  $\times$  means coefficientwise multiplication of grid functions  $(u \times v)_k = u_k v_k$ . Unfortunately, we do not have the same luck as in the case of differentiation. We have the following result.

**THEOREM 3.12.** *Suppose  $(h, g, \tilde{h}, \tilde{g})$  is a perfect reconstruction filter bank, and  $\tilde{g}$  has  $N$  zero-moments. Then a necessary and sufficient condition for*

$$(3.33) \quad (H^*)^j(\tilde{H}^j[x^k]_h \times \tilde{H}^j[x^s]_h) = [x^{k+s}]_h \quad \forall j \quad \forall k \quad \forall s : s + k \leq N_2 \leq N - 1$$

is that

$$(3.34) \quad \sigma_p(\tilde{h}) = \sqrt{2} \left( \frac{\sigma_1(\tilde{h})}{\sqrt{2}} \right)^p, \quad p = 1, \dots, N_2 - 1.$$

A necessary and sufficient condition for

$$(3.35) \quad (H^*)^j([x^k]_{2^j h} \times \tilde{H}^j[x^s]_h) = [x^{k+s}]_h \quad \forall j \quad \forall k \quad \forall s : s + k \leq N_2 \leq N - 1$$

is that

$$(3.36) \quad \sigma_p(\tilde{h}) = 0, \quad p = 1, \dots, N_2 - 1.$$

*Proof.* We start with  $\tilde{H}^j[x^k]_h \times \tilde{H}^j[x^s]_h$  and prove  $(\Rightarrow)$ . The same reasoning as in Theorem 3.5 shows that it is the same to say that

$$(\tilde{H}[x^k]_h \times \tilde{H}[x^s]_h) = \tilde{H}[x^{k+s}]_h,$$

i.e.,

$$\frac{1}{2} \sum_{n, n_2} \tilde{h}_{n-2p} \tilde{h}_{n_2-2p} n^s n_2^k = \frac{1}{\sqrt{2}} \sum_n \tilde{h}_{n-2p} n^{s+k},$$

or

$$(3.37) \quad \frac{1}{2} \sum_{n, n_2} \tilde{h}_n \tilde{h}_{n_2} (n + 2p)^s (n_2 + 2p)^k = \frac{1}{\sqrt{2}} \sum_n \tilde{h}_n (n + 2p)^{s+k}.$$

This is trivial for  $s = 0$ ,  $k < N$ . For  $s > 0$  we write the right-hand side of (3.37) as

$$\begin{aligned} & \frac{1}{\sqrt{2}} \sum_n \tilde{h}_n (n + 2p)^{s+k} \frac{1}{\sqrt{2}} \sum_{n_2} \tilde{h}_{n_2} \\ &= \frac{1}{2} \sum_{n, n_2} \tilde{h}_n \tilde{h}_{n_2} (n_2 + 2p + n - n_2)^s (n + 2p)^k \\ &= \frac{1}{2} \sum_{n, n_2} \tilde{h}_n \tilde{h}_{n_2} (n_2 + 2p)^s (n + 2p)^k \\ &+ \frac{1}{2} \sum_{t=1}^s \binom{s}{t} \sum_{n, n_2} \tilde{h}_n \tilde{h}_{n_2} (n - n_2)^t (n_2 + 2p)^{s-t} (n + 2p)^k, \end{aligned}$$

so (3.37) is equivalent to

$$(3.38) \quad \sum_{t=1}^s \binom{s}{t} \sum_{n, n_2} \tilde{h}_n \tilde{h}_{n_2} (n - n_2)^t (n_2 + 2p)^{s-t} (n + 2p)^k = 0 \quad \forall p.$$

From the zero-moment conditions of  $\tilde{h}$  we can write this as

$$(3.39) \quad 2 \sum_{t=1}^s \binom{s}{t} \left( \frac{\sigma_1(\tilde{h})}{\sqrt{2}} \right)^t \left( \sum_{v=0}^t \binom{t}{v} (-1)^v \right) \left( \frac{\sigma_1(\tilde{h})}{\sqrt{2}} + 2p \right)^{s-t} \left( \frac{\sigma_1(\tilde{h})}{\sqrt{2}} + 2p \right)^k = 0.$$

This can be seen by binomially expanding the powers and replacing all sums over  $n, n_2$ , with the moment conditions. But (3.39) is trivially satisfied as the sum over  $v$  equals zero. So the  $(\Rightarrow)$  part of the theorem when multiplying two wavelet expansions is proved.

For the  $(\Leftarrow)$  part, we choose  $s = 1$ ,  $p = 0$ , and  $k > 0$  as the first number for which  $\sigma_{k+1}(\tilde{h}) \neq \sqrt{2}(\sigma_1(\tilde{h})/\sqrt{2})^{k+1}$ . We then have that the left-hand side of (3.38) equals

$$\sum_{n, n_2} \tilde{h}_n \tilde{h}_{n_2} (n - n_2) n^k = \sqrt{2} \sigma_{k+1}(\tilde{h}) - \sigma_k(\tilde{h}) \sigma_1(\tilde{h}) \neq 0,$$

so (3.38) is not satisfied, and as  $H^*$  is injective for polynomials,  $(H^*)(\tilde{H}[x]_h \times \tilde{H}[x^M]_h) \neq [x^{M+1}]_h$ .

We now turn to the second part of the theorem. Equation (3.35) is equivalent to

$$[x^k]_{2h} \times \tilde{H}[x^s]_h = \tilde{H}[x^{k+s}]_h \quad \forall k \forall s : s + k \leq N_2 \leq N - 1,$$

i.e.,

$$(3.40) \quad \frac{1}{\sqrt{2}} \sum_n \tilde{h}_n (n + 2p)^s (2p)^k = \frac{1}{\sqrt{2}} \sum_n \tilde{h}_n (n + 2p)^{s+k} \quad \forall p.$$

The left-hand side of (3.40) is

$$\frac{1}{\sqrt{2}} \sum_n \tilde{h}_n \sum_{j=0}^s \binom{s}{j} n^j (2p)^{s-j} (2p)^k = \sum_{j=0}^s \binom{s}{j} \sigma_j(\tilde{h}) (2p)^{s+k-j},$$

so  $\sigma_j = 0$ ,  $j = 1, \dots, N_2 - 1$  is necessary and sufficient for (3.40) to be satisfied.  $\square$

The filter bank  $\beta_2$  will therefore fail to multiply linear functions correctly on lower levels. To get higher-order results in the multiplication, we must have filter banks where  $\tilde{h}$  satisfies zero-moment conditions. This was noted in [4], [5] for the Galerkin method, where coiflets were used. The filter banks in Table 2.3 satisfy zero-moment conditions, but these “waste” zero-moments on  $h$ , and the filter banks in Table 2.4 are preferably used. When  $\tilde{h}$  corresponds to a scaling function  $\tilde{\varphi} \in L^2$ , we know from [29] that

$$(3.41) \quad \Sigma_p(\tilde{\varphi}) = \frac{1}{2^p - 1} \sum_{i=1}^p \binom{p}{i} \sigma_i(\tilde{h}) \Sigma_{p-i}(\tilde{\varphi}),$$

so (3.34) is equivalent to

$$(3.42) \quad \Sigma_p(\tilde{\varphi}) = (\Sigma_1(\tilde{\varphi}))^p, \quad p = 1, \dots, N_2 - 1,$$

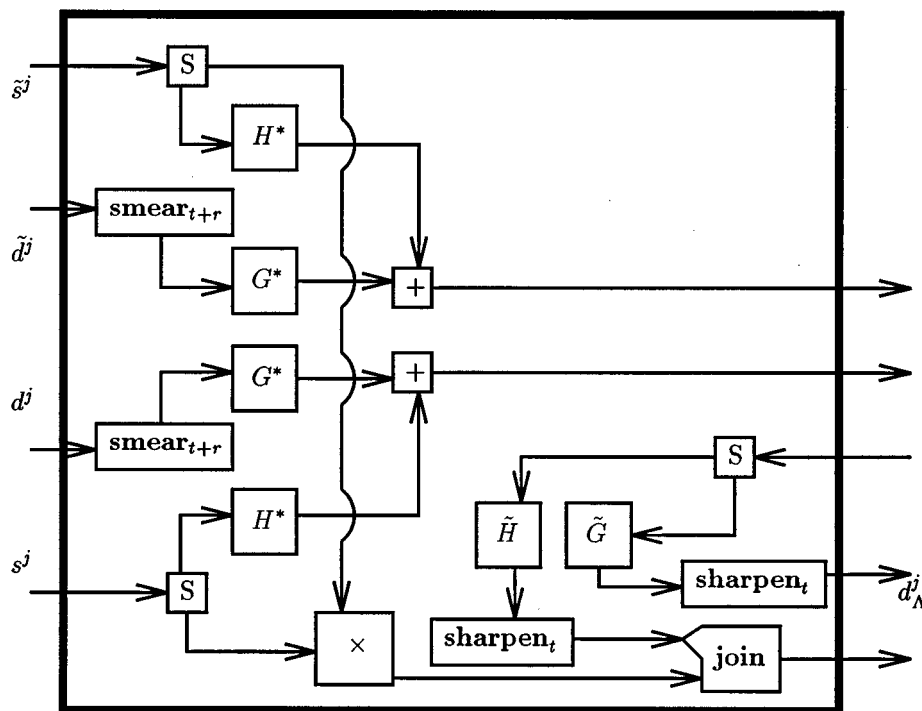


FIG. 22. One step in the  $\Lambda_X$ -cycle.

and also to

$$(3.43) \quad \Sigma_p(\tilde{\varphi}(\cdot + \Sigma_1(\tilde{\varphi}))) = 0, \quad p = 1, \dots, N_2 - 1.$$

With this in mind we now turn to the algorithm for multiplying two functions.

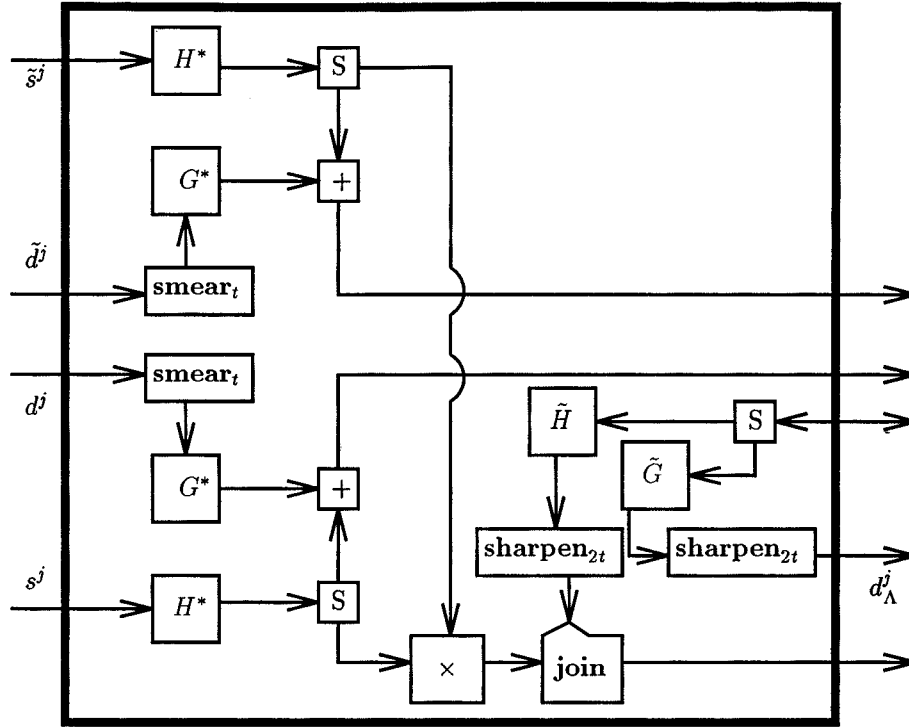
The multiplication of a grid function  $u = [f]_{2^{-n}}$  and a sparse wavelet representation  $\tau_\epsilon$  will lead to the same method as for differentiation, except that the convolution with  $p(h)$  is replaced by coefficientwise multiplication with  $[f]_{2^{-n+j}}$ . When multiplying *two* sparse wavelet representations, we define a step in the  $\Lambda_X$ - and  $\tilde{\Lambda}_X$ -cycles in Figures 22 and 23. The total  $\Lambda$ -cycle is shown in Figure 24. The total  $\tilde{\Lambda}$ -cycle can also be defined in the same manner as for differentiation.

For numerical tests we use the same test function as in the previous section  $f = \sin(2\pi x) + e^{-20000(x-0.5)^2}$  and square it, giving the function in Figure 25. We compare the  $\Lambda$ -cycle with transforming all the way back to the finest scale, giving

$$(3.44) \quad e_\Lambda = \mathcal{F}^{-k} \mathcal{T}_\epsilon \Lambda_{\times^2} \mathcal{T}_\epsilon \mathcal{F}^k [f]_{2^{-n}} - [f^2]_{2^{-n}},$$

$$(3.45) \quad e_a = \mathcal{F}^{-k} \mathcal{T}_\epsilon \mathcal{F}^k (\times^2) \mathcal{F}^{-k} \mathcal{T}_\epsilon \mathcal{F}^k [f]_{2^{-n}} - [f^2]_{2^{-n}}.$$

In our first example we use the filter bank  $\beta_2$  and choose  $n = 10$ ,  $k = 4$ ,  $\epsilon = 10^{-4}$ . The error is shown in Figure 26, plotted in  $\log_{10}$  scale. We see here that, in contrast to differentiation, the method of transforming back everywhere gives a smaller error than the  $\Lambda$ -cycle, which seems to introduce a low-frequency error. This comes from the lack of zero-moments for  $\tilde{h}$ . As the total error is a combination of the error in the truncation step  $\mathcal{T}_\epsilon$  and in the  $\Lambda$ -cycle, this effect will show only for “small”  $\epsilon$ .

FIG. 23. One step in the  $\tilde{\Lambda}_X$ -cycle.

In our second example, we use the  $\delta_3$  filter bank, with the same parameters as in the previous example. The result is shown in Figure 27. We see here that we have improved the result, and there is no large low-frequency component in the error. The results with this filter bank when truncating and differentiating with  $\Lambda$ -cycles are similar to the result when using the  $\beta_2$  filter bank.

In the following two sections we shall use the filter bank method and the  $\Lambda$ - and  $\tilde{\Lambda}$ -cycles to solve PDEs in one and two dimensions.

**4. Numerical experiments in one dimension.** The methods described in the previous section are efficient for differentiating and multiplying functions in the functions spaces  $\mathcal{L}^{s,m}$ , i.e., functions that are smooth nearly everywhere but with a few small regions with sharp gradients. Here efficient means that for  $F$ , an operator involving multiplication and differentiation,

- $u$  can be represented by  $u_\epsilon$ , with  $N_s \ll N$  coefficients, where  $N = 2^n$  is the number of coefficients on the finest level;
- there is an approximation of  $F$ ,  $\tilde{F}$  such that  $\tilde{F}u_\epsilon$  can be evaluated in  $\mathcal{O}(N_s)$  arithmetic operations;
- the error  $e = Fu - \tilde{F}u_\epsilon$  is small, typically  $\|e\| = \mathcal{O}(N^{-\alpha} + \epsilon^\gamma)$ , with  $\alpha > 1$ ,  $\gamma > 0$ .

A method that satisfies these properties will be well suited to use in an explicit solver for a hyperbolic PDE. We shall now see how well the  $\Lambda$ - and  $\tilde{\Lambda}$ -cycles perform.



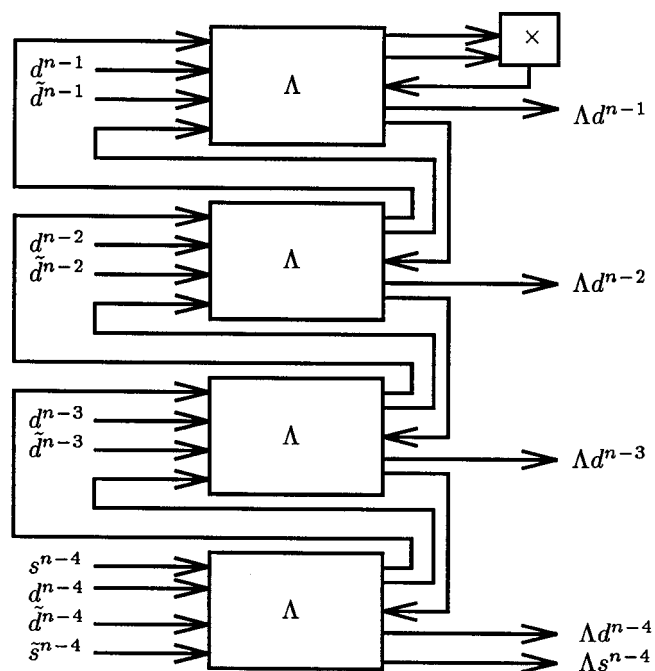


FIG. 24. Total sparse multiplication with  $\Lambda_\times$ -cycle.

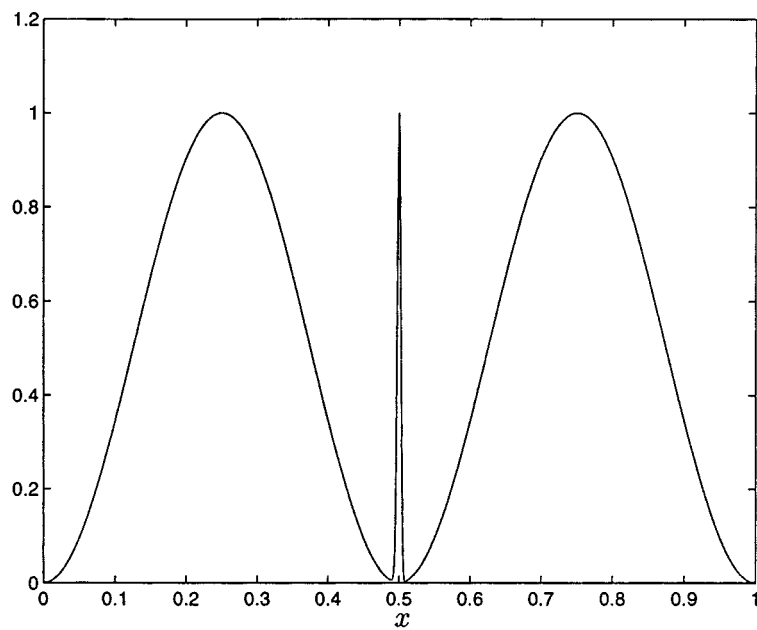


FIG. 25. Square of  $f = \sin(2\pi x) + e^{-20000(x-0.5)^2}$ .

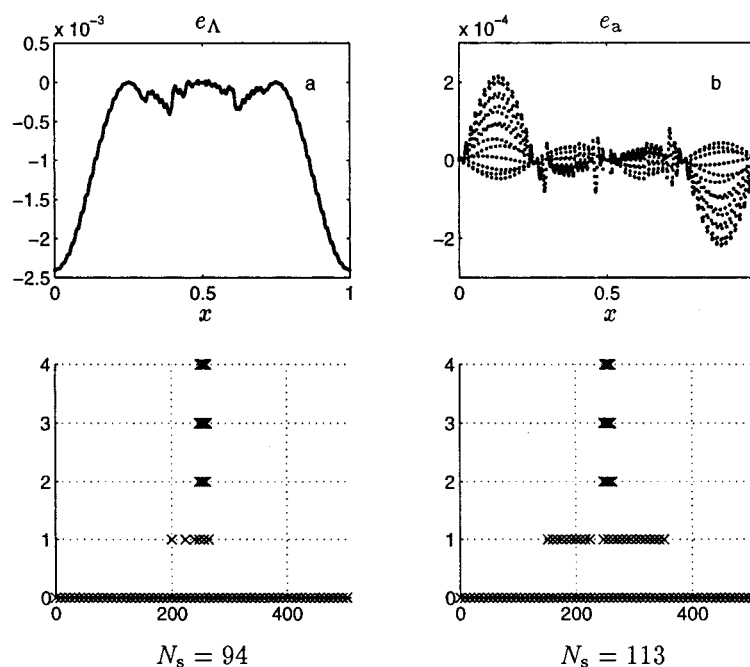


FIG. 26. Error and sparse representation for two methods to perform multiplication (a)  $\Lambda$ -cycle and (b) full inverse transform, with  $\beta_2$  filter bank,  $\epsilon = 10^{-4}$ .

We consider a general initial value problem, with periodic BCs,

$$(4.1) \quad \begin{cases} u_t = P(d/dx, x, t, u), \\ u(0, t) = u(1, t), \\ u(x, 0) = u_0(x), \\ 0 \leq x \leq 1, t \geq 0. \end{cases}$$

We suppose  $P$  can be Taylor expanded in each argument and that the series can be truncated (without changing the solution “too much”), giving a right-hand side of the equation that can be evaluated by repeated multiplication and differentiation. We discretize in space with  $\Delta x = 2^{-n}$  and use an explicit time-marching scheme with time step  $\Delta t$ . We will use the fourth-order classical Runge–Kutta method in all examples. This gives us an approximation

$$\tilde{u}_k^n \approx u(n\Delta t, k\Delta x).$$

The time step has been chosen somewhat smaller than optimal in all examples, although the  $\Lambda$ - and  $\tilde{\Lambda}$ -cycles do not seem to decrease the optimal time step. This has earlier been noted for similar methods, e.g., in [20].

When implementing the  $\Lambda$ -cycles on a computer, we used linked lists to represent the sparse grid functions. As there is a considerable amount of overhead involved in finding coefficients, etc., blocks of  $n_b = 2^l$  coefficients (with  $l$  typically between 2 and 5) were grouped together, giving a representation as in Figure 28. By choosing a power of 2 for  $n_b$  we make the transformation between block level and coefficient level fast. The block versions of  $\mathcal{T}_\epsilon$ ,  $\ast^{\text{sparse}}$ , **smear**, **sharpen**, **join**, are defined in

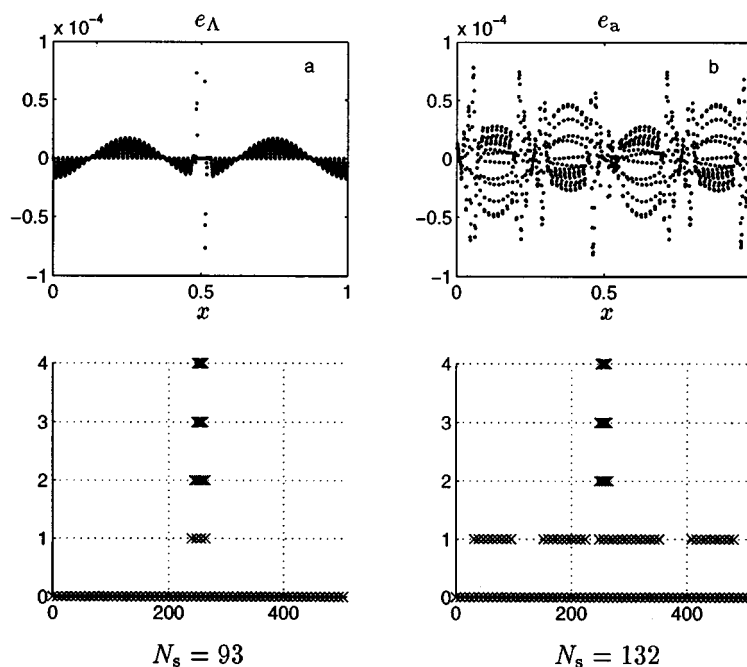


FIG. 27. Error and sparse representation for two methods to perform multiplication (a)  $\Lambda$ -cycle and (b) full inverse transform, with  $\delta_3$  filter bank,  $\epsilon = 10^{-4}$ .

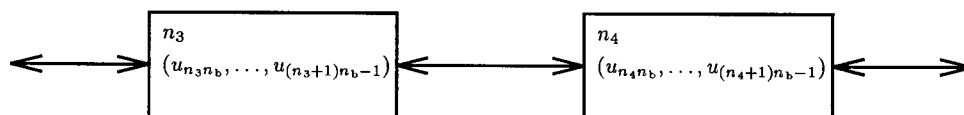


FIG. 28. Block linked list representation of sparse grid functions.

the same manner as previously. We redefine  $N_s$  to mean the number of *blocks* in the thresholded expansion (instead of the number of *coefficients*).

In [18] a tree representation was chosen for the transformed signal. The advantage of this representation is that it is easy to find coefficients that have overlapping support on different levels. Conversely, extra work is needed to construct such a tree representation. In [18] all calculations were performed with the wavelet representation, and coefficients on different levels were needed at the same time. With the  $\Lambda$ -cycle, however, all calculations are performed with coefficients on the same level, so the links between different levels are not needed, and the linked list representation will perform better.

The one-dimensional test problems we study are the advection equation and the Burgers equation.

#### 4.1. The advection equation. We solve

$$(4.2) \quad \begin{cases} u_t = u_x, & 0 \leq x \leq 1, \quad t \geq 0, \\ u(x, 0) = u_0(x), & u = u(x, t), \quad u(0, t) = u(1, t). \end{cases}$$

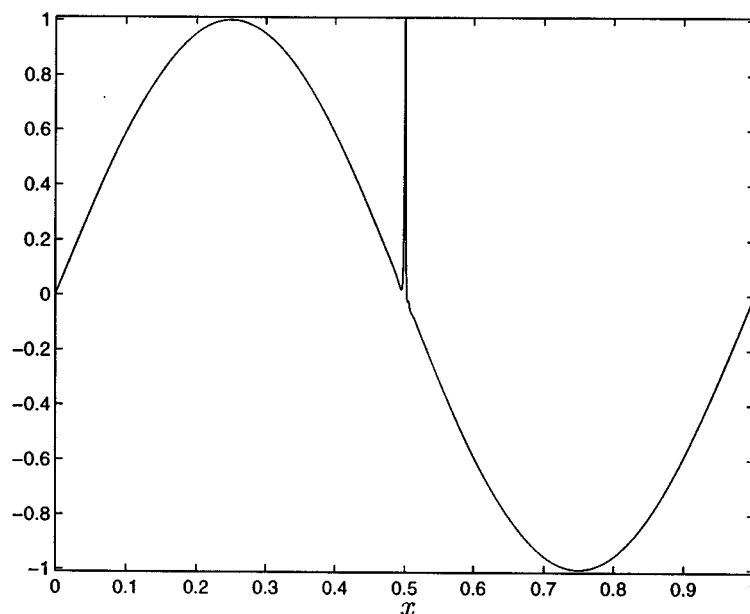


FIG. 29. Approximated solution at  $t = 1$  when solving  $u_t = u_x$  with  $\tilde{\Lambda}$ -cycle and  $\beta_2$  filter bank.

We choose the initial function  $u_0(x) = \sin(2\pi x) + e^{-5 \times 10^5 (x-0.5)^2}$ , which is the same as in the previous examples but with a sharper “spike.” The solution is a translation of the initial condition  $u(x, t) = u_0((x + t) \bmod 1)$ . We choose  $\Delta t = 10^{-4}$  and step forward one period to  $t = 1$ . The parameters are  $n = 13$ ,  $k = 4$ ,  $\epsilon = 6 \times 10^{-5}$ , and we use the  $\beta_2$  filter bank. Finally, we use the second-order approximation of  $d/dx$ ,  $d(h) = D_0 = \frac{1}{2h}(1, 0, -1)$ . The number of blocks in the approximated solution (each block containing  $n_b = 8$  coefficients) was never more than 80, out of a total of 1024, so the compression factor was about 13. In Figure 29 we see the approximated solution. We see here that the “spike” has been resolved, and it has not decreased in magnitude. In Figure 30 we show the solution and the approximation around the “spike.” In Figure 31 we compare the error (plotted in  $\log_{10}$  scale) of the  $\Lambda$ -cycle with solving without truncation. We see that the untruncated method performs somewhat better, even in max-norm. When we solved with  $n = 12$ , the untruncated method did not manage to resolve the spike, and the error was large. The same pattern appeared in the other numerical examples. *When the untruncated method solved the problem properly, the  $\Lambda$ -cycle (and, for nonlinear problems, the  $\tilde{\Lambda}$ -cycle) usually did so too, with much fewer coefficients but with a somewhat larger error.*

We next sharpen the “spike” more and choose  $u_0(x) = \sin(2\pi x) + e^{-4 \times 10^6 (x-0.5)^2}$ . For this problem it was difficult for  $\beta_2$  and  $D_0$  to resolve the “spike” properly, and some high frequency errors appeared, ruining the sparse representation of the signal. When we chose  $\beta_4$  and the fourth-order scheme  $d(h) = \frac{1}{h}(-\frac{1}{12}, \frac{2}{3}, 0, -\frac{2}{3}, \frac{1}{12})$ , this problem disappeared. The parameters were  $n = 15$ ,  $k = 6$ ,  $\Delta t = 2.5 \times 10^{-5}$ ,  $\epsilon = 10^{-6}$ . The approximation at  $t = 1$  is shown in Figure 32 and the error in Figure 33. We see that 85 blocks out of a total of 2048 are used, giving a compression rate of 24.1.

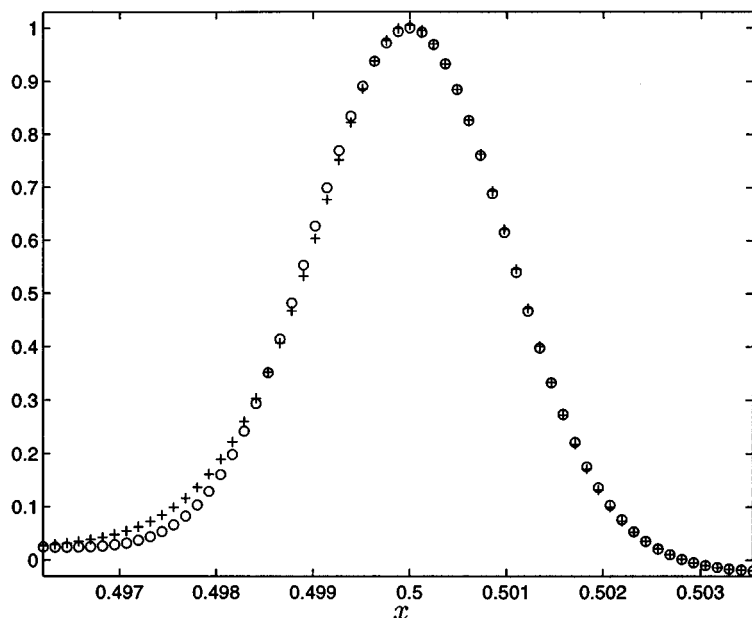


FIG. 30. Solution “o,” and approximated solution “+,” around “spike” at  $t = 1$  when solving with  $\Lambda$ -cycle and  $\beta_2$  filter bank.

**4.2. The Burgers equation.** We want to solve the nonlinear Burgers equation,

$$(4.3) \quad \begin{cases} u_t + uu_x = \mu u_{xx}, & 0 \leq x \leq 1, \quad t \geq 0, \mu > 0, \\ u(x, 0) = \sin(2\pi x), & u = u(x, t), \quad u(0, t) = u(1, t). \end{cases}$$

The exact solution is shown in Figure 34 for viscosity  $\mu = 10^{-4}$ . The difficulty with this equation, apart from being nonlinear, is the very sharp gradient that forms at  $x = 0.5$ , although the solution is smooth in other parts of the region.

One of the conclusions in [18] was that it is difficult to perform multiplication efficiently in the wavelet domain, and this was one reason for constructing the filter bank method and the  $\Lambda$ -cycle that uses point value information. We therefore examine how well the filter bank method solves the problem. We choose the  $\beta_2$  filter bank  $n = 13$ ,  $k = 7$ ,  $\epsilon = 10^{-4}$ , and the fourth-order finite difference filters  $d(h) = \frac{1}{h}(-\frac{1}{12}, \frac{2}{3}, 0, -\frac{2}{3}, \frac{1}{12})$ ,  $d_{d^2/dx^2}(h) = \frac{1}{12h^2}(-1, 16, -30, 16, -1)$ . The time step is set to  $\Delta t = 2.5 \times 10^{-5}$ . The approximated solution at  $t = 0.1$ ,  $t = 0.2$ ,  $t = 0.5$  is shown in Figure 35.

The block size is  $n_b = 8$ . Here we see how the number of blocks go from 8 on scale  $\frac{1}{64}$  at  $t = 0.1$ , to 22 with finest resolution  $\frac{1}{8192}$  at  $t = 0.2$ . This increase in resolution is performed automatically by the  $\tilde{\Lambda}$ -cycle. In Figure 36 we show the solution near the sharp gradient. There are about 10 coefficients in the shock. The compression factor is  $1024/22 = 46.5$ . The exact solution can be found by the Cole–Hopf transformation [9], [19]. The error in max-norm at  $t = 0.2$  is  $\|e\|_\infty = 0.0628$ . The filter bank method with  $\tilde{\Lambda}$ -cycle thus performs very well for the Burgers equation, even with the filter banks without zero-moments for  $\tilde{h}$ , zero-moments that would increase the accuracy in the multiplication step.

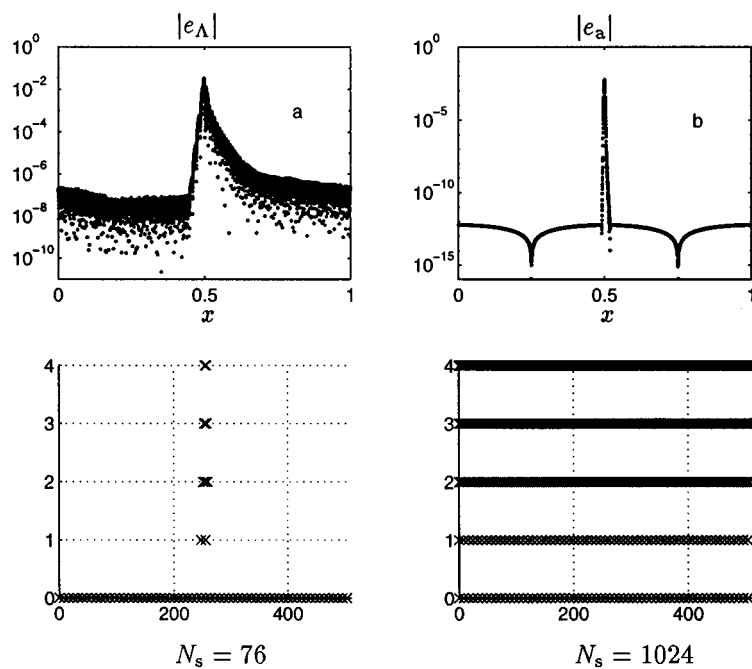


FIG. 31. Error and sparse representation for two methods of solving  $u_t = u_x$  (a)  $\Lambda$ -cycle with  $\beta_2$  filter bank and (b) untruncated.

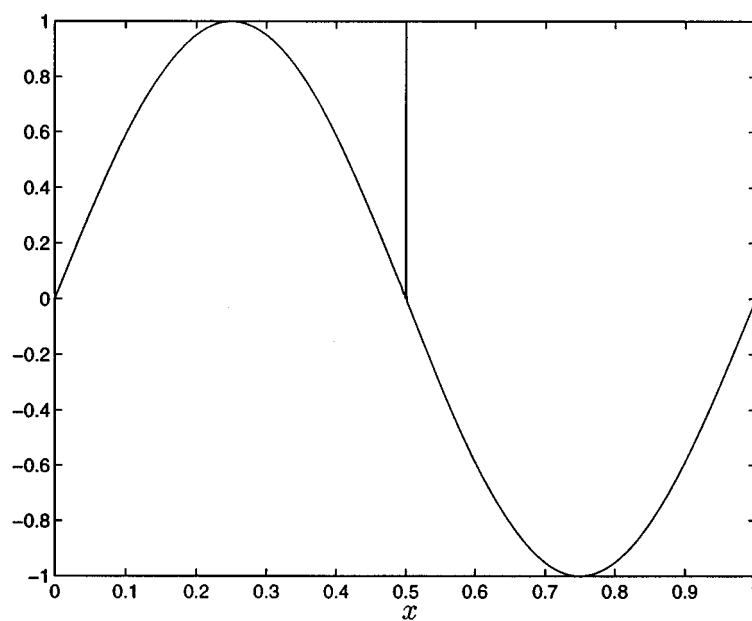


FIG. 32. Approximated solution at  $t = 1$  when solving  $u_t = u_x$  with  $\Lambda$ -cycle and  $\beta_4$  filter bank.

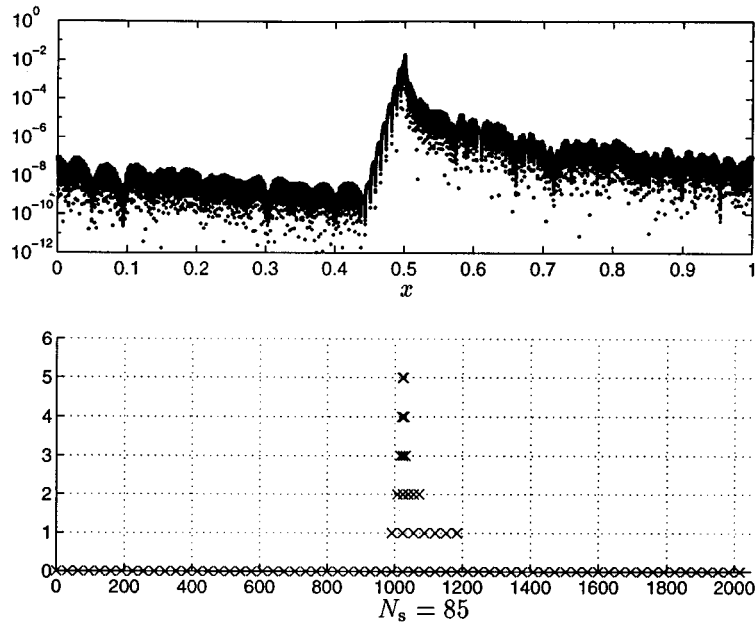


FIG. 33. Error and representation when solving  $u_t = u_x$  with  $\Lambda$ -cycle and  $\beta_4$  filter bank.

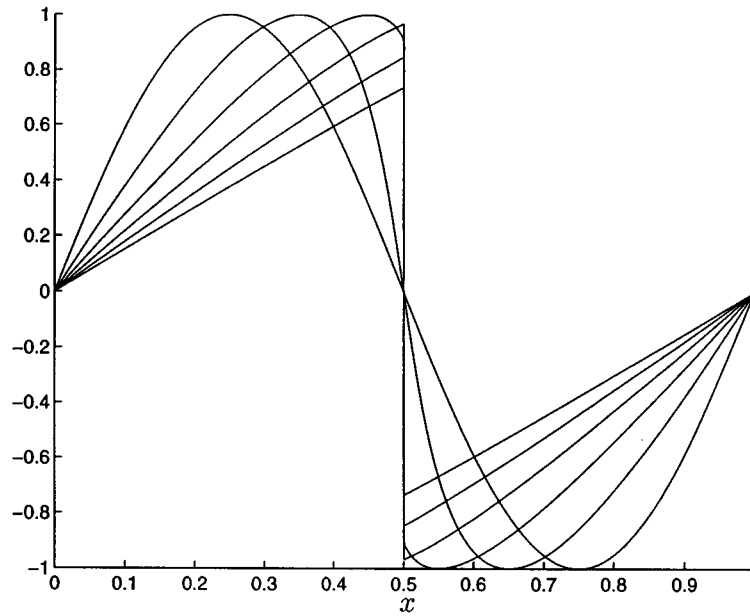


FIG. 34. Solution to the Burgers equation with viscosity  $\mu = 10^{-4}$  at times  $t = 0.0, 0.1, 0.2, 0.3, 0.4, 0.5$ .

**5. Numerical experiments in two dimensions.** The extension of continuous wavelet theory to two dimensions, keeping the locality in both directions, is done by

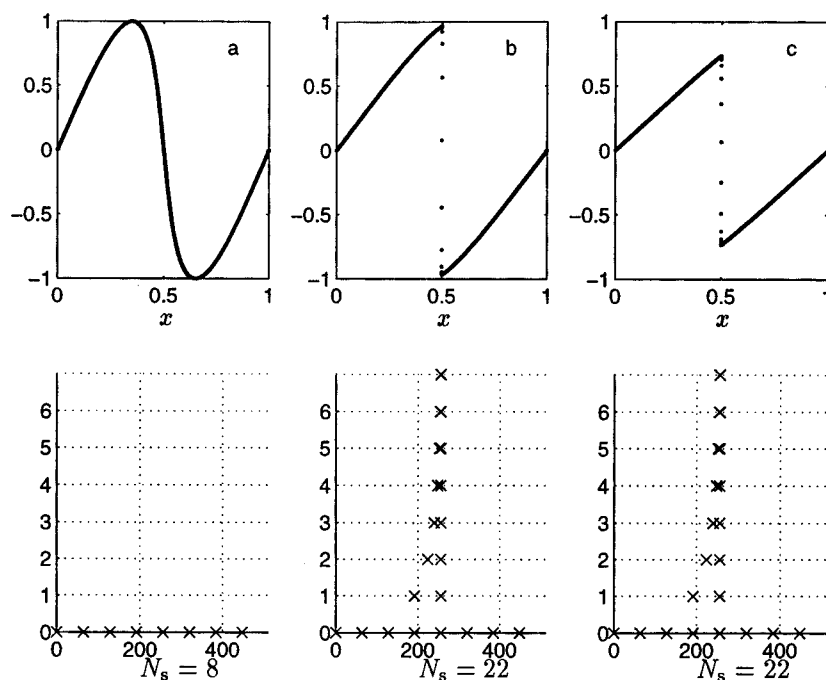


FIG. 35. Approximated solution at (a)  $t = 0.1$ , (b)  $t = 0.2$ , (c)  $t = 0.5$  when solving the Burgers equation with viscosity  $\mu = 10^{-4}$  using the  $\tilde{\Lambda}$ -cycle, filter bank  $\beta_2$ , and  $\epsilon = 5 \times 10^{-5}$ .

the tensor product spaces [11],

$$\begin{aligned} \mathcal{V}_{j+1} &= \mathcal{V}_j \oplus \mathcal{W}_j, \\ \mathcal{V}_j &= V_j \otimes V_j, \\ (5.1) \quad \mathcal{W}_j &= (V_j \otimes W_j) \oplus (W_j \otimes W_j) \oplus (W_j \otimes V_j). \end{aligned}$$

The filter bank transform will be on two-dimensional grid functions  $u : \mathbb{Z} \times \mathbb{Z} \rightarrow \mathbb{C}$ , and convolution, up-, and downsampling can now be made in either the  $x$  or  $y$  direction. The discrete two-dimensional filter bank transform relies on the identity

$$\begin{aligned} (5.2) \quad u &= (h^*_x \uparrow_x h^*_y \uparrow_y \downarrow_y \tilde{h}^*_y \downarrow_x \tilde{h}^*_x + h^*_x \uparrow_x g^*_y \uparrow_y \downarrow_y \tilde{g}^*_y \downarrow_x \tilde{h}^*_x \\ &+ g^*_x \uparrow_x h^*_y \uparrow_y \downarrow_y \tilde{h}^*_y \downarrow_x \tilde{g}^*_x + g^*_x \uparrow_x g^*_y \uparrow_y \downarrow_y \tilde{g}^*_y \downarrow_x \tilde{g}^*_x) u. \end{aligned}$$

The sparse linked list representation is not easily generalized to more than one dimension. A “naive” approach, viewing a two-dimensional sparse grid function as a long one-dimensional grid function,  $u = (u_{m_x(u),:}, u_{m_x(u)+1,:}, \dots, u_{M_x(u),:})$  as in Figure 37a, will lead to unacceptable work to find neighbors in the  $y$  direction. On the contrary, a definition that has an ordering in both  $x$  and  $y$  directions, as in Figure 37b, leads to difficulties in how to treat grid functions where there are diagonal blocks involved. The way chosen in this paper is to see a two-dimensional grid function as an array of one-dimensional grid functions, as in Figure 37c. This structure makes the  $\Lambda$ - and  $\tilde{\Lambda}$ -cycles easy to generalize to two dimensions. For example, we have  $\mathbf{smear}_{2d} u = \mathbf{smear}_x \mathbf{smear}_y u$ ,  $\mathbf{sharpen} u = \mathbf{sharpen}_x \mathbf{sharpen}_y u$ . The extension to more than two dimensions is immediate.



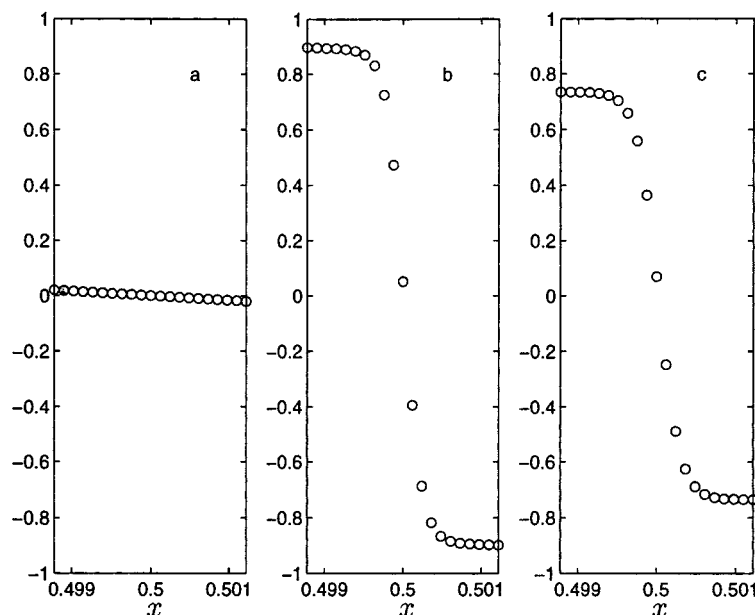


FIG. 36. Approximated solution near  $x = 0.5$  at (a)  $t = 0.1$ , (b)  $t = 0.2$ , (c)  $t = 0.5$  when solving the Burgers equation with viscosity  $\mu = 10^{-4}$  using the  $\tilde{\Lambda}$ -cycle, filter bank  $\beta_2$ , and  $\epsilon = 5 \times 10^{-5}$ .

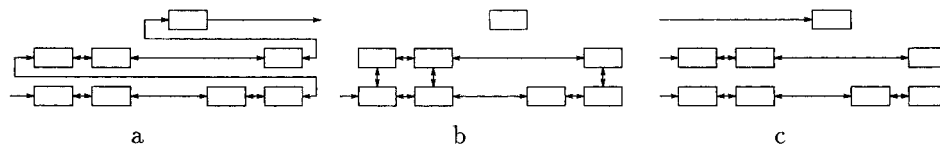


FIG. 37. Linked list representations of two-dimensional grid functions.

The general two-dimensional PDE we want to solve is

$$(5.3) \quad \begin{cases} u_t = P(d/dx, d/dy, x, y, t, u), \\ u(x, y, 0) = u_0(x, y), \\ u(0, y, t) = u(1, y, t), \\ u(x, 0, t) = u(x, 1, t), \\ 0 \leq x \leq 1, 0 \leq y \leq 1, t \geq 0, \end{cases}$$

and we make the same assumption as in the one-dimensional case, i.e., that we can Taylor expand  $P$  and get an approximation that can be evaluated by repeated multiplication and differentiation. We also permit systems, i.e.,  $u = (u_1, \dots, u_n)^T$ .

**5.1. The advection equation.** We solve

$$(5.4) \quad \begin{cases} u_t = u_x + u_y, \\ u(x, 0, t) = u(x, 1, t), \\ u(0, y, t) = u(1, y, t), \\ u(x, y, 0) = u_0(x, y), \\ 0 \leq x \leq 1, 0 \leq y \leq 1, t \geq 0, \end{cases}$$

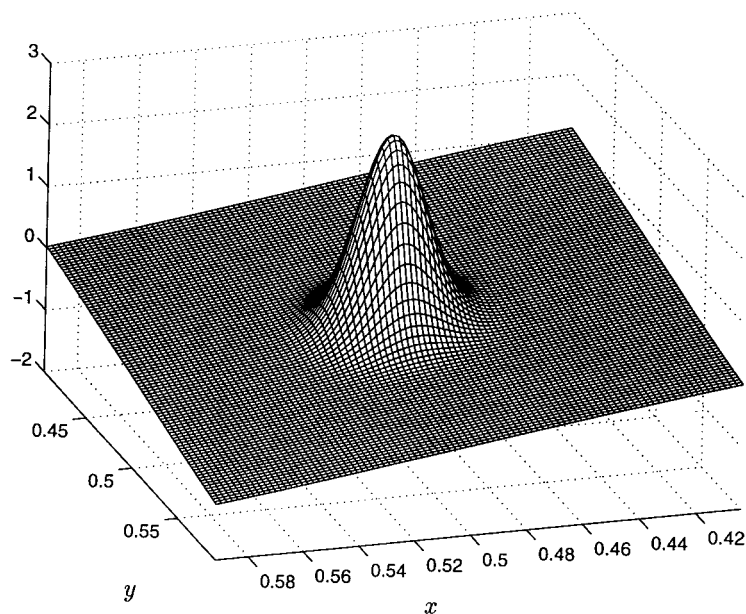


FIG. 38. Approximated solution at  $t = 1$  when solving  $u_t = u_x + u_y$  with  $\beta_2$  filter bank.

with initial function  $u_0(x, y) = \sin(2\pi x) + \sin(2\pi y) + 3e^{-2500((x-0.5)^2 + (y-0.5)^2)}$ . The solution to this equation is  $u(x, y, t) = u_0((x+t) \bmod 1, (y+t) \bmod 1)$ . We choose the  $\beta_2$  filter bank, the parameters  $n = 9$ ,  $k = 3$ ,  $\Delta t = 10^{-3}$ ,  $\epsilon = 10^{-5}$ , and the fourth-order centered finite difference scheme. The approximated solution at  $t = 1$  is shown in Figure 38, and the error is shown in Figure 39. We have  $\|e\|_\infty = 6.1 \times 10^{-3}$ . The total number of blocks used is 828 out of 16,384 ( $n_b = 4$ ), giving a compression factor of 19.8. These blocks are shown in Figure 40, where we have separated the four components of the transformed signal, e.g.,  $V \otimes W = \downarrow_y \check{h} *_{\check{y}} \downarrow_x \check{g} *_{\check{x}} (\downarrow_y \check{h} *_{\check{y}} \downarrow_x \check{h} *_{\check{x}})^{j-1} u$ .

We next choose the “nastier” initial function  $u_0(x, y) = \sin(2\pi x) + \sin(2\pi y) + 3e^{-10000((x-0.5)^2 + (y-0.5)^2)}$  and solve the PDE with the  $\beta_4$  filter bank and the parameters  $n = 10$ ,  $k = 4$ ,  $\Delta t = 5 \times 10^{-4}$ ,  $\epsilon = 10^{-5}$ , and the fourth-order centered finite difference scheme. The approximated solution at  $t = 1$  is shown in Figure 41 and the error in Figure 42. We have  $\|e\|_\infty = 0.02488$ . The total number of blocks used is 148 out of 16,384 ( $n_b = 8$ ), giving a compression factor of 110.7. These blocks are shown in Figure 43.

The “tail” behind the “spike” appeared in several of the tests and is due to the thresholding not killing enough high frequencies. *It usually did not grow as the time increased.* We see that the filter bank method automatically adapts to a grid that is locally refined in both the  $x$  and  $y$  direction, a problem that becomes “messy” when dealing with locally refined grids and finite differences. Furthermore, changing the order of the method is easily done by changing the filter bank.

**5.2. The equation  $u_t = 2\pi(y - 0.5)u_x - 2\pi(x - 0.5)u_y$ .** The next equation is linear but with variable coefficients

$$(5.5) \quad \begin{cases} u_t = 2\pi(y - 0.5)u_x - 2\pi(x - 0.5)u_y, \\ u(x, y, 0) = e^{-6400((x-0.25)^2 + (y-0.5)^2)}, \\ (x, y) \in \mathbb{R}^2, t \geq 0. \end{cases}$$

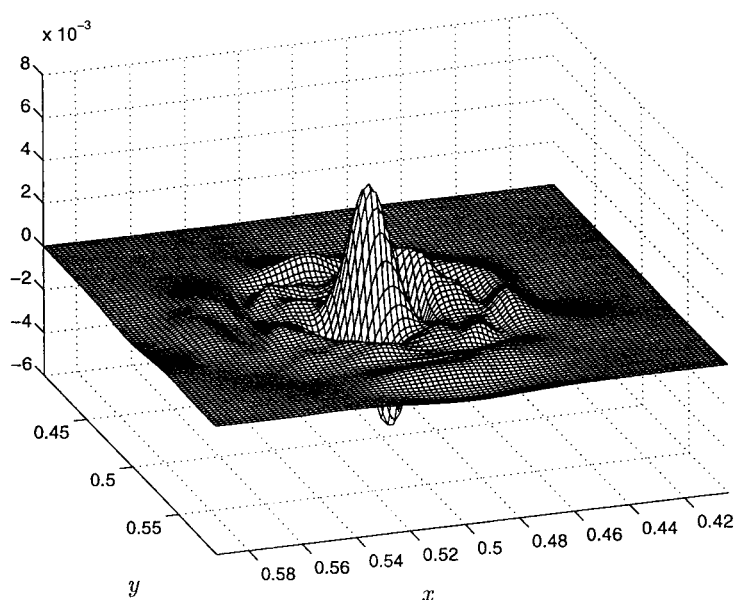


FIG. 39. Error at  $t = 1$  when solving  $u_t = u_x + u_y$  with  $\beta_2$  filter bank.

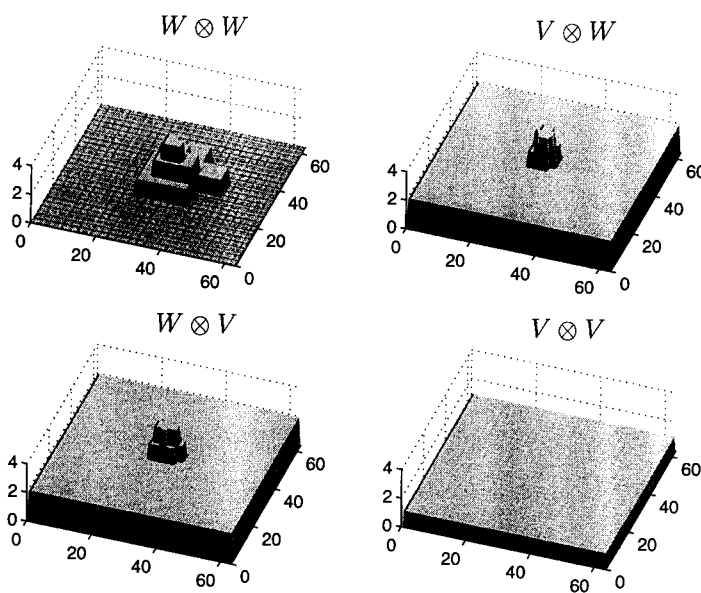


FIG. 40. Sparse representation at  $t = 1$  when solving  $u_t = u_x + u_y$  with  $\beta_2$  filter bank.

The solution will rotate around  $(0.5, 0.5)$  with period 1, so  $u(x, y, 1) = u(x, y, 0)$ . The  $\beta$  filter banks will be inadequate for this equation because of the low order of accuracy in the multiplication. Therefore, we choose the  $\delta_3$  filter bank. We choose the fourth-order centered finite difference scheme  $n = 10$ ,  $k = 3$ ,  $\Delta t = 8 \times 10^{-4}$ ,  $\epsilon = 3 \times 10^{-5}$ .

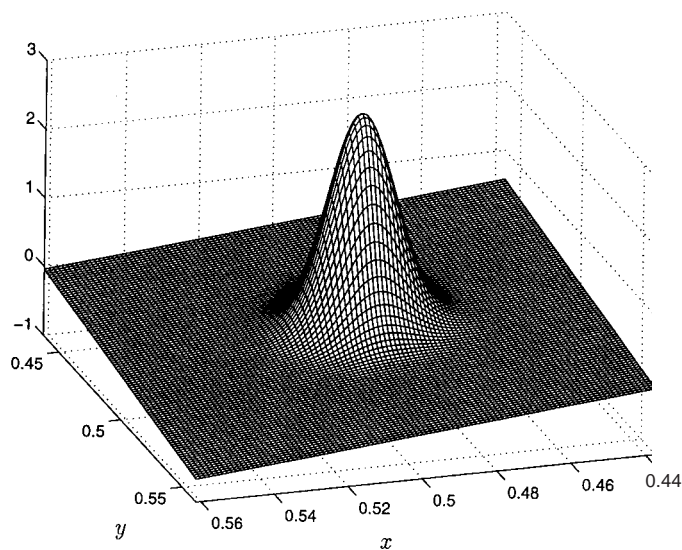


FIG. 41. *Approximated solution at  $t = 1$  when solving  $u_t = u_x + u_y$  with  $\beta_4$  filter bank.*

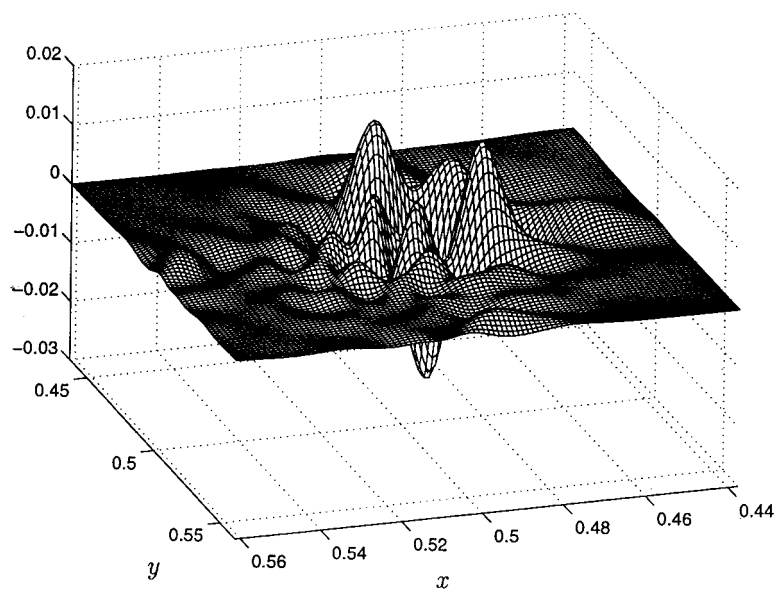


FIG. 42. *Error at  $t = 1$  when solving  $u_t = u_x + u_y$  with  $\beta_4$  filter bank.*

The approximated solution around the “spike” at  $t = 1$  is shown in Figure 44, and the error is shown in Figure 45. The max-norm of the error is  $\|e\|_\infty = 0.00973$ . In Figure 46 we see the representation at  $t = 1$ . We have 57 blocks out of a total of 4,096, giving a compression factor of 71.9. As there is no low-frequency signal around the “spike,” there are regions where there is no grid at all (shown in the  $V \otimes V$  part of Figure 46).

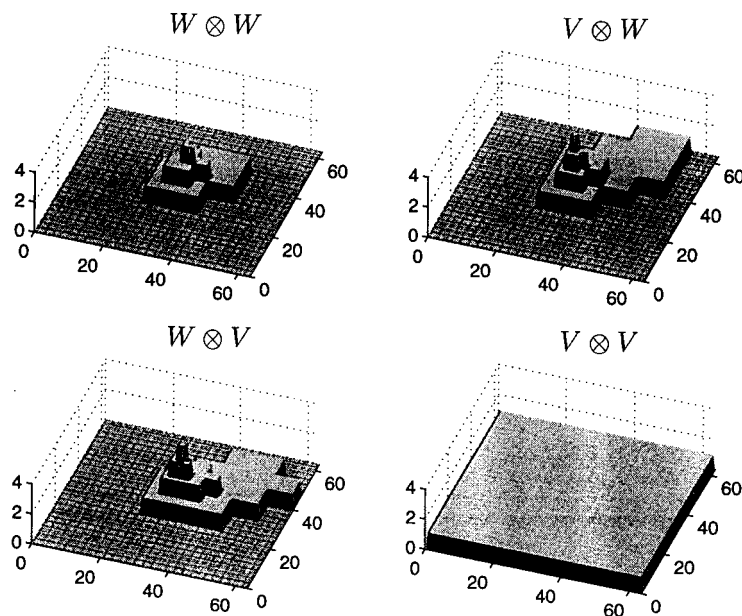


FIG. 43. Sparse representation at  $t = 1$  when solving  $u_t = u_x + u_y$  with  $\beta_4$  filter bank.

**5.3. The bidimensional Burgers equation.** Our last example is the bidimensional Burgers equation

$$(5.6) \quad \begin{cases} u_t = -(\nabla u)u + \mu \Delta u, \\ u(x, 0, t) = u(x, 1, t), \\ u(0, y, t) = u(1, y, t), \\ u(x, y, 0) = (\sin(2\pi(x + y)), \sin(2\pi(x + y)))^T, \\ 0 \leq x \leq 1, 0 \leq y \leq 1, t \geq 0, \end{cases}$$

where  $u = (u_1, u_2)^T$ . This equation was studied in [27]. With these initial conditions, the solution is a  $45^\circ$  rotation of the solution to the one-dimensional equation

$$(5.7) \quad u(x, y, t) = (u_{1d}(x + y, 2t), u_{1d}(x + y, 2t))^T,$$

but, because of the rotation, the equation is fully bidimensional. The initial condition is shown in Figure 47, and the solution at  $t = 0.1$  for  $\mu = 10^{-2}$  is shown in Figure 48.

We choose  $\mu = 5 \times 10^{-4}$ ,  $n = 11$ ,  $k = 5$ ,  $\epsilon = 8 \times 10^{-5}$ , the  $\delta_3$  filter bank, the fourth-order finite difference schemes, and the time step  $\Delta t = 10^{-4}$ . The approximated solution near  $(x, y) = (0.25, 0.25)$  at  $t = 0.2$  is shown in Figure 49, and the representation is shown in Figure 50. In Figure 51 we see the approximated solution around  $(0.25, 0.25)$  in the diagonal direction  $(x, y) = \gamma(\alpha) = \alpha(1, 1)$  compared with the exact solution. The error along this path is  $\|e|_\gamma\|_\infty = 6.07 \times 10^{-4}$ . This improvement compared with the one-dimensional case comes from the use of the  $\delta$ -filter bank. The number of blocks is 2,296 out of a total of 65,536 ( $n_b = 8$ ), giving a compression factor of 28.5. Thus, the filter bank method, as in the one-dimensional case, works very well for this nonlinear problem.

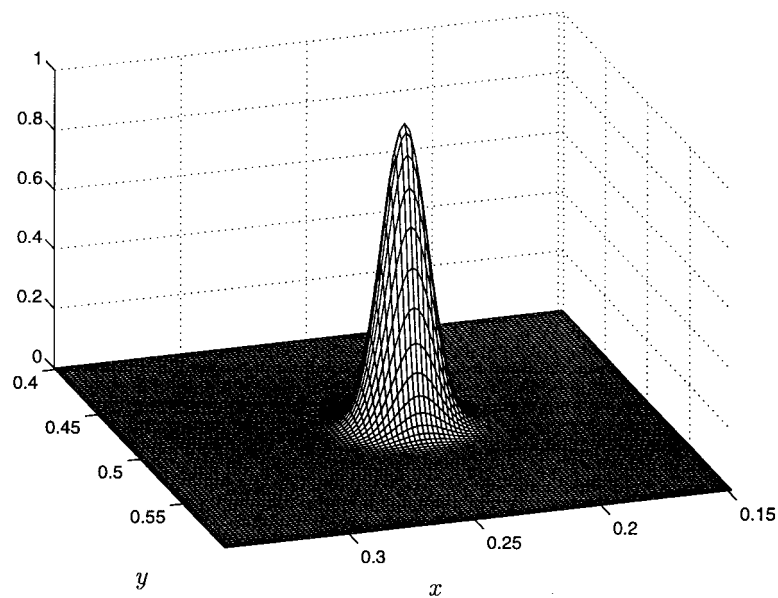


FIG. 44. *Approximated solution at  $t = 1$  when solving  $u_t = 2\pi(y - 0.5)u_x - 2\pi(x - 0.5)u_y$  with  $\delta_3$  filter bank.*

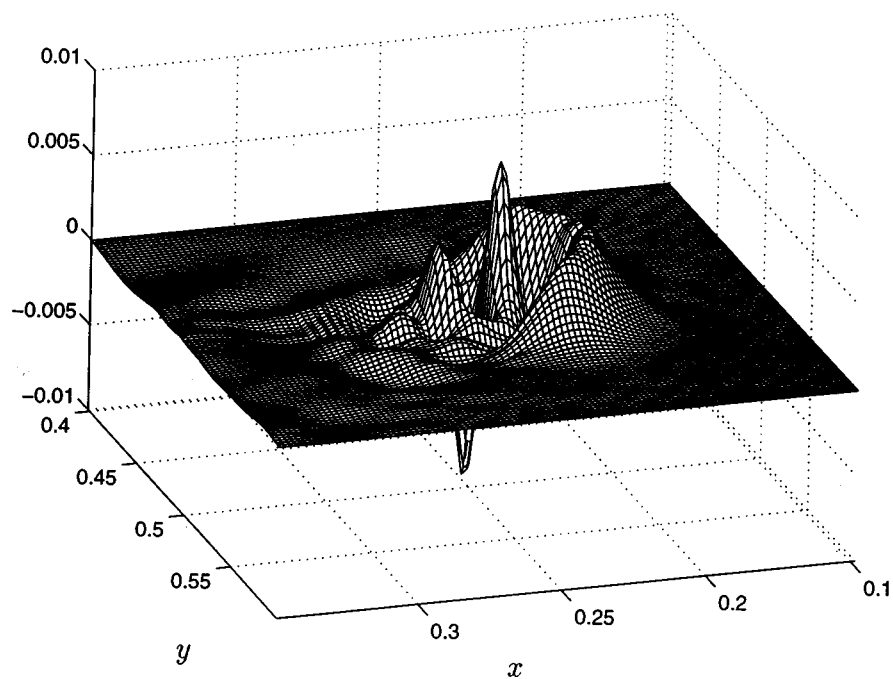


FIG. 45. *Error at  $t = 1$  when solving  $u_t = 2\pi(y - 0.5)u_x - 2\pi(x - 0.5)u_y$  with  $\delta_3$  filter bank.*

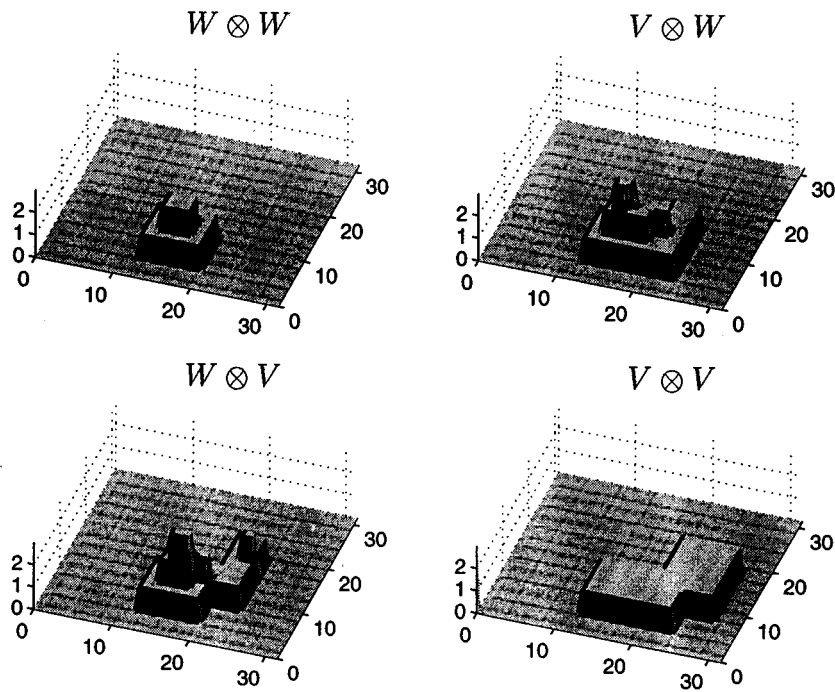


FIG. 46. Sparse representation at  $t = 1$  when solving  $u_t = 2\pi(y - 0.5)u_x - 2\pi(x - 0.5)u_y$  with  $\delta_3$  filter bank.

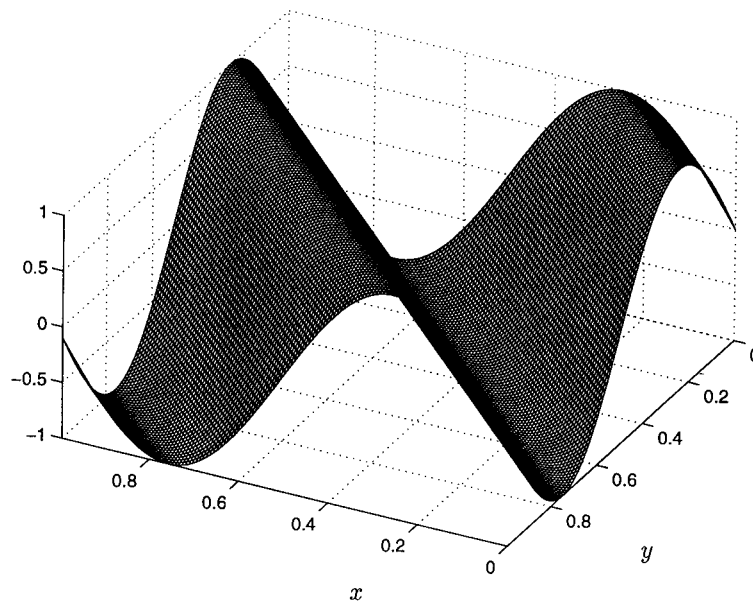


FIG. 47. Initial condition for the bidimensional Burgers equation.

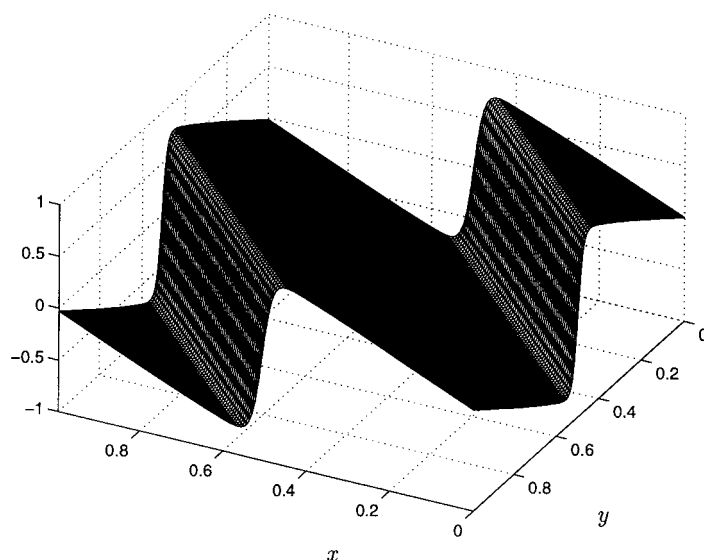


FIG. 48. Solution at  $t = 0.1$  for the bidimensional Burgers equation with  $\mu = 10^{-2}$ .

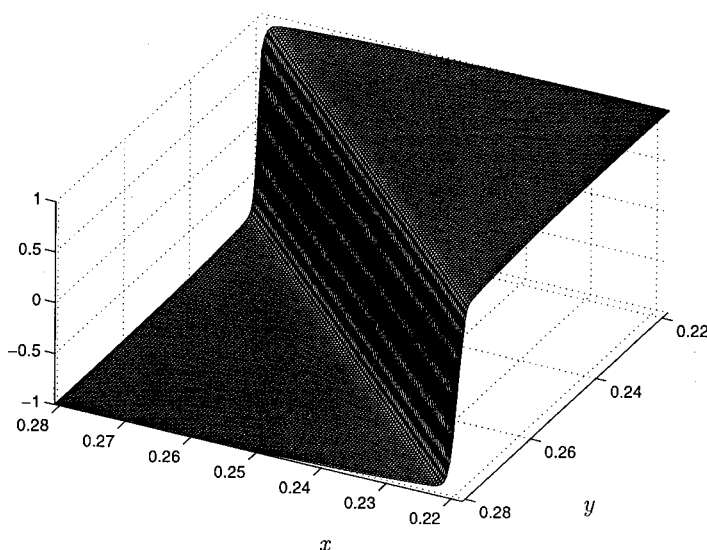


FIG. 49. Approximated solution at  $t = 0.1$  for the bidimensional Burgers equation around  $(0.25, 0.25)$  with  $\mu = 5 \times 10^{-4}$ ,  $\epsilon = 8 \times 10^{-5}$  when solved with  $\delta_3$  filter bank  $\tilde{\Lambda}$ -cycle.

**6. Conclusions.** We have developed a method for solving hyperbolic initial value problems in an adaptive multilevel manner. By working with perfect reconstruction filter banks and sampled values of the function, we reduced the length of the filters involved substantially and decreased the error in the method. The filters used in this paper, the  $\beta$  and  $\delta$  filter banks, do not correspond to any wavelets in  $L^2$ , and the modest price paid is extra growth factors in the error and number of coefficients.



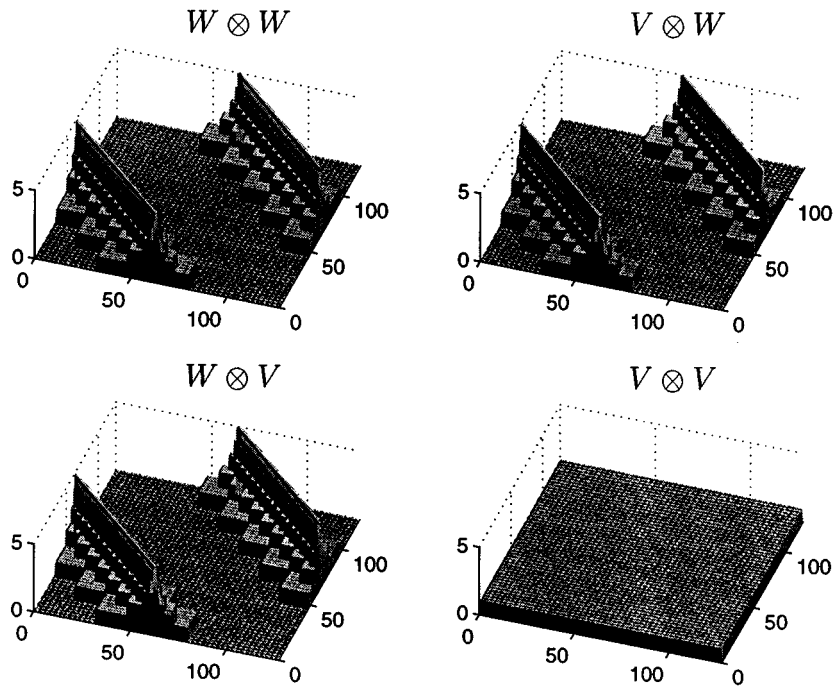


FIG. 50. Sparse representation at  $t = 0.1$  when solving the bidimensional Burgers equation with  $\delta_3$  filter bank,  $\tilde{\Lambda}$ -cycle, viscosity  $\mu = 5 \times 10^{-4}$ ,  $\epsilon = 8 \times 10^{-5}$ .

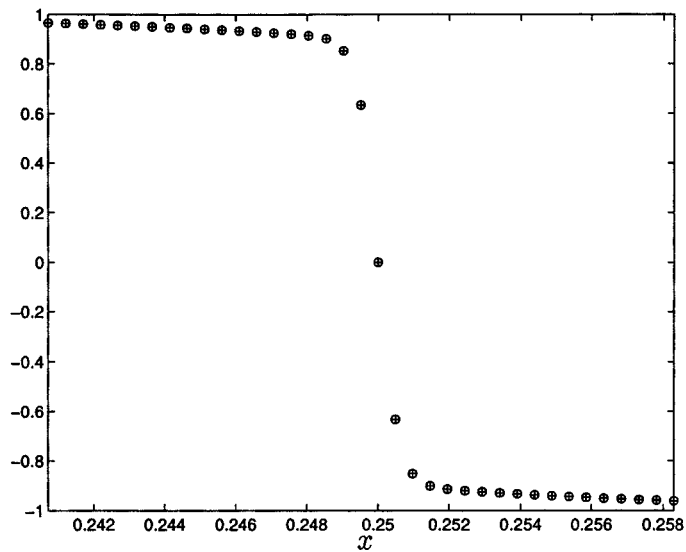


FIG. 51. Approximated solution "o," and exact solution "+," near  $(0.25, 0.25)$  along path  $(x, y) = \alpha(1, 1)$  at  $t = 0.1$  when solving the bidimensional Burgers equation with viscosity  $\mu = 5 \times 10^{-4}$  using the  $\tilde{\Lambda}$ -cycle, filter bank  $\delta_3$ , and  $\epsilon = 8 \times 10^{-5}$ .

The filter bank method provides simple rules of how to use locally refined grids in an arbitrary number of dimensions. The only parameter needed is the truncation  $\epsilon$ . Furthermore, changing the order of the method is easy by switching filter banks.

The  $\Lambda$ - and  $\tilde{\Lambda}$ -cycles could be used for other local operators than multiplication and differentiation, although the filter banks used might have to be tailored to satisfy some extra conditions (as was the case when switching from  $\beta$  to  $\delta$  filter banks for multiplication).

The cycles could also be modified to allow adaptive time steps as in [1], although it is an open question whether the increased complexity in the algorithm is outweighed by the larger time steps.

The compression factors of 13–110 in the test problems make the method highly interesting for large scale problems where the  $\tilde{\Lambda}$ -cycle seems robust enough to handle a large number of different situations. Adding boundary conditions should impose no major difficulties, due to the attractive combination of using short filters and working partly in the physical domain.

#### REFERENCES

- [1] E. BACRY, S. MALLAT, AND G. PAPANICOLAOU, *A wavelet based space-time adaptive numerical method for partial differential equations*, Math. Model Numer. Anal., 26 (1992), p. 793.
- [2] T. P. BARNWELL AND M. J. T. SMITH, *Exact reconstruction techniques for tree-structured subband coders*, IEEE Trans. Acoust. Speech Signal Process., 34 (1986), pp. 434–441.
- [3] S. BERTOLUZZA AND G. NALDI, *A Wavelet Collocation Method for the Numerical Solution of Partial Differential Equations*, Technical Report 887, Istituto di Analisi Numerica del Consiglio Nazionale Delle Ricerche, Italy, 1993.
- [4] G. BEYLKIN AND J. KEISER, *An adaptive pseudo-wavelet approach for solving nonlinear partial differential equations*, Wavelet Anal. Appl., 6 (1997), pp. 137–197.
- [5] G. BEYLKIN AND J. KEISER, *On the adaptive numerical solution of nonlinear partial differential equations in wavelet bases*, J. Comput. Phys., 132 (1997), pp. 233–259.
- [6] A. COHEN, *Ondelettes, analyses multirésolutions et filtres miroir en quadrature*, Ann. Inst. H. Poincaré Anal. Non Linéaire, 7 (1990), pp. 439–459.
- [7] A. COHEN, *Biorthogonal wavelets*, Wavelet Anal. Appl., 2 (1992), pp. 123–152.
- [8] A. COHEN, I. DAUBECHIES, AND J. C. FEAVEU, *Biorthogonal bases of compactly supported wavelets*, Comm. Pure Appl. Math., 45 (1992), pp. 485–560.
- [9] J. D. COLE, *On a quasilinear parabolic equation occurring in aerodynamics*, Quart. App. Math., 9 (1951), pp. 225–236.
- [10] I. DAUBECHIES, *Orthonormal bases of compactly supported wavelets*, Comm. Pure Appl. Math., 41 (1988), pp. 909–996.
- [11] I. DAUBECHIES, *Ten Lectures on Wavelets*, SIAM, Philadelphia, 1992.
- [12] I. DAUBECHIES, *Orthonormal bases of compactly supported wavelets II. Variations on a theme*, SIAM J. Math. Anal., 24 (1993), pp. 499–519.
- [13] G. DESLAURIERS AND S. DUBUC, *Symmetric iterative interpolation processes*, Constr. Approx., 5 (1989), pp. 49–68.
- [14] T. EIROLA, *Sobolev characterization of solutions of dilation equations*, SIAM J. Math. Anal., 23 (1992), pp. 1015–1030.
- [15] G. FIX AND G. STRANG, *A Fourier analysis of the finite element variational method*, in Constructive Aspects of Functional Analysis, Edizione Cremonese, Rome, 1973, pp. 795–846.
- [16] J. FRÖHLICH AND K. SHNEIDER, *An Adaptive Wavelet-Vaguelette Algorithm for the Solution of Nonlinear PDEs*, Konrad-Zuse-Zentrum, Berlin, Germany.
- [17] A. HARTEN, *Multiresolution algorithms for the numerical solution of hyperbolic conservation laws*, Comm. Pure Appl. Math., 48 (1995), pp. 1305–1342.
- [18] M. HOLMSTRÖM AND J. WALDÉN, *Adaptive wavelet methods for hyperbolic PDEs*, J. Sci. Comput., 13 (1998), pp. 19–49.
- [19] E. HOPF, *The partial differential equation  $u_t + uu_x = \mu u_{xx}$* , Comm. Pure Appl. Math., 3 (1950), pp. 201–230.
- [20] L. JAMESON, *On the Wavelet Optimized Finite Difference Method*, Technical Report 94-9, NASA, Langley Research Center, Hampton, VA, 1994.

- [21] L. JAMESON, *A Wavelet-Optimized, Very High Order Adaptive Grid and Order Numerical Method*, Technical Report ICASE, 96-30, NASA, Langley Research Center, Hampton, VA, 1996.
- [22] W. M. LAWTON, *Tight frames of compactly supported affine wavelets*, J. Math. Phys., 31 (1990), pp. 1899–1901.
- [23] W. M. LAWTON, *Necessary and sufficient conditions for constructing orthonormal wavelet bases*, J. Math. Phys., 32 (1991), pp. 57–61.
- [24] S. MALLAT, *Multiresolution approximations and wavelet orthonormal bases of  $L^2(\mathbb{R})$* , Trans. Amer. Math. Soc., 315 (1989), pp. 69–87.
- [25] Y. MEYER, *Wavelets and Operators*, Cambridge University Press, London, 1992.
- [26] F. MINTZER, *Filters for distortion-free two-band multirate filter banks*, IEEE Trans. Acoust. Speech Signal Process., 33 (1985), pp. 626–630.
- [27] P.J. PONENTI AND J. LIANDRAT, *Numerical Algorithms Based on Biorthogonal Wavelets*, Technical Report ICASE 96-13, NASA, Langley Research Center, Hampton, VA, 1996.
- [28] W. SWELDENS AND R. PIESENS, *Asymptotic error expansion of wavelet approximations of smooth functions II*, Numer. Math., 68 (1994), pp. 377–401.
- [29] W. SWELDENS AND R. PIESENS, *Quadrature formulae and asymptotic error expansions for wavelet approximations of smooth functions*, SIAM J. Numer. Anal., 31 (1994), pp. 1240–1264.
- [30] O. V. VASILYEV AND S. PAOLUCCI, *A dynamically adaptive multilevel wavelet collocation method for solving partial differential equations in a finite domain*, J. Comput. Phys., 125 (1996), pp. 498–512.
- [31] O. V. VASILYEV, S. PAOLUCCI, AND M. SEN, *A multilevel wavelet collocation method for solving partial differential equations in a finite domain*, J. Comput. Phys., 120 (1995), pp. 33–47.