Wavelet Analysis and Ocean Modeling: A Dynamically Adaptive Numerical Method "WOFD-AHO"

LELAND JAMESON AND TORU MIYAMA

International Pacific Reseach Center, University of Hawaii, Honolulu, Hawaii

(Manuscript received 18 December 1998, in final form 24 May 1999)

ABSTRACT

Wavelet analysis provides information on the energy present at various scales and locations throughout a computational domain. This information is precisely the information that is needed to define the appropriate gridpoint densities and the appropriate numerical order to resolve the physics at hand in the computationally most efficient manner. Here a two-dimensional version of the numerical method known as the Wavelet Optimized Finite Difference Method (WOFD) is introduced to a model problem in oceanography. WOFD is a completely dynamically adaptive numerical method that has the ability to focus on small-scale physics throughout the computational domain as the scale of the physics gets smaller and as structures are transported across the domain. In this manner, WOFD applies the computational effort where it is needed without overcomputing in the regions of the domain where the physics is somewhat smooth and perhaps more linear. The version of WOFD used here is such that both the spatial and temporal orders will be fixed at four. In the areas of oceanography and climate modeling, order four can be considered high order and therefore this two-dimensional version will be called the Adaptive High Order (AHO) version of WOFD, or simply WOFD-AHO.

1. Introduction

Wavelet transforms provide information about a function or dataset with respect to scale and location in contrast to Fourier transforms, which provide a one-parameter family of coefficients representing the global frequency content. In numerical computations, one often encounters computational data that have a variety of scales at different locations throughout the computation domain. Furthermore, the data can be dynamically moving throughout the computational domain. One might conjecture that such a computational environment could best be computed with a wavelet basis. We introduce here a two-dimensional version of a numerical method that we have named the Wavelet Optimized Finite Difference Adaptive High Order (WOFD-AHO) method. WOFD-AHO dynamically moves with the data and focuses in on the data at the appropriate scale to resolve whatever scales are present. Furthermore, WOFD-AHO is fourth order in both space and time. This gives a far more accurate solution than leapfrog style methods, which are commonly used in oceanography.

In addition, these days there is a tremendous focus on building adaptive numerical methods in the computational sciences. The simple reason is that interesting physical problems rarely occur with uniform scale features throughout the computational domain. We can, of course, apply uniform grid techniques to such problems with a tremendous waste of computational effort. In fact, it is likely that a well-written adaptive scheme run on a five-year-old, or older, supercomputer can outperform a nonadaptive scheme on the current generation of supercomputers. It is this kind of savings that must be obtained in order to conduct leading edge oceanography.

Ocean dynamics include many processes that occur over a wide range of spatial scales. While the scale of the general circulation is $\sim 10^4$ km, synoptic mesoscale variability is on the order of less than 10 km to several tens of kilometers. To understand the interaction of such wide-range-scale phenomena, numerical models that include all scales have been sought.

One way to achieve this is a very high-resolution grid model. Usually the smallest grid size is determined primarily by computer limitations, rather than by the flow physics. Although this is beginning to become better with rapid progress in computer technology (Beckmann et al. 1994; Kagimoto and Yamagata 1997; McClean et al. 1997), it is not sufficient at this point. In addition, the interesting mesoscale eddies are often seen in a limited area (e.g., western boundary regions). In that case, an evenly spaced fine resolution is a luxury.

An alternative to overcome some of these difficulties is a nested grid model. By including a nested structure, a selected region can be examined at higher resolution

Corresponding author address: Dr. Leland Jameson, IPRC, University of Hawaii, 2525 Correa Rd., Honolulu, HI 96822.
E-mail: ljameson@soest.hawaii.edu

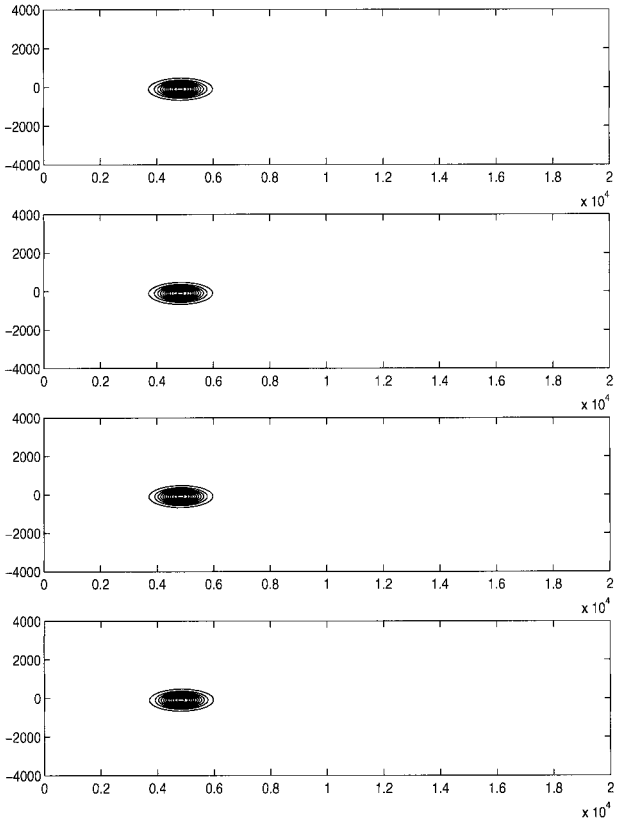

FIG. 1. The initial condition for variable h at four different wavelet thresholds. Contour interval is 5 m.

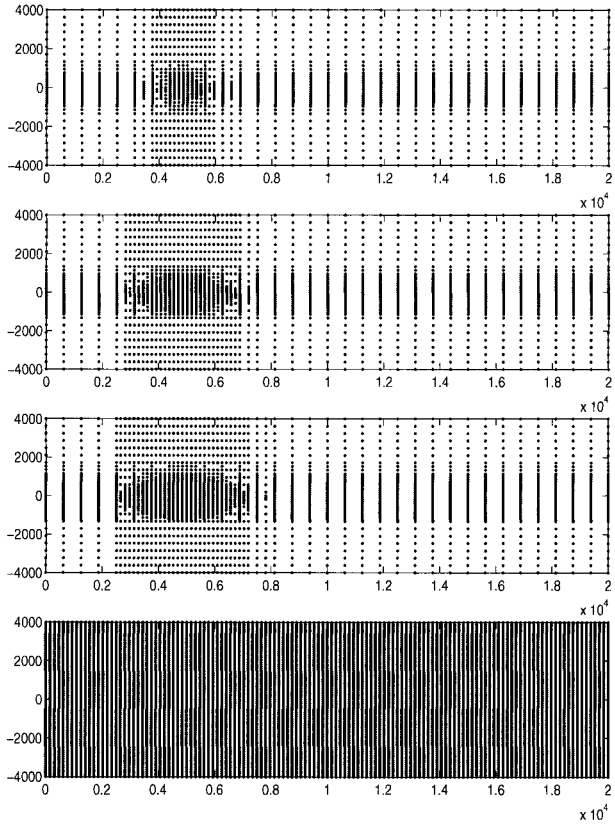

FIG. 2. The initial wavelet generated grids at four different wavelet thresholds. From top to bottom the four different wavelet thresholds are 0.001, 0.0001, 0.00001, and 0.

while the large-scale flow can be simulated with a lower resolution. This technique is widely used in meteorology (Zhang et al. 1986). There are also some applications in oceanography (Fox and Maskell 1995; Spall and Holland 1991; Oey and Chen 1992; Ginis et al. 1998). However, nested grid models often introduce an artificial boundary into the domain requiring that information be passed from the the coarse to the fine grid and back again. We believe that the introduction of this artificial boundary should generally be avoided.

Now a few comments on wavelet-based numerical methods. Generally, one can classify ''wavelet methods'' as either a collocation type or a Galerkin type. Note that the terms collocation and Galerkin are completely independent from the wavelet. These terms define the manner in which the continuous problem as defined by the partial differential equations is projected into the discrete world to be executed on a computer. One can think of the computational parameters in a wavelet collocation method as the point values of the computational variables in the physical space. Likewise, one can think of the computational parameters in a wavelet Galerkin method as the point values in the transform space, in this case the wavelet transform space. In either case, one is working with N real numbers. The best way to evolve these N real numbers in a wavelet or multiresolution framework is a matter of debate. This paper will introduce WOFD-AHO in which wavelets are used only for grid refinement.

2. An overview of wavelet-based numerical methods

First we begin with an overview of the important features of wavelet-based numerical methods.

The key strength of wavelet methods is data compression. An efficient basis is one in which a given set of data can be represented with as few basis elements as possible. So that a sine wave can be represented, without error, with one basis element if the basis is composed of sine waves, whereas a sine wave would require a large number of wavelets without ever decreasing the error to zero. So, are wavelets better than sine waves for computational fluid dynamics? This depends on the problem. If during a computation one always has smooth and periodic data, then sines and cosines are probably appropriate. On the other hand, if the computational domain contains very fine structure in one part of the domain and smooth structure in the remainder of the domain, then it is possible that wavelets

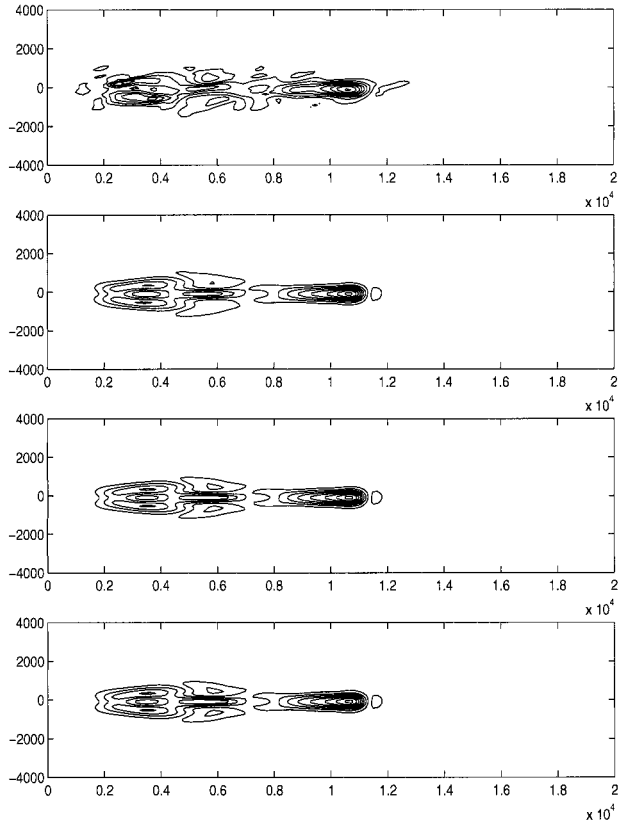

FIG. 3. Four flows at four wavelet thresholds at day 35. Contour interval is 2 m.

TABLE 1. Parameters in the test cases.

Case 1		
(Lx, Ly)	Model region	20 000 km × 8000 km
$\Delta x, \Delta y$	Grid resolution	156 km, 95 km
Δt	Time step	3050 s
g'	Reduced gravity	0.049 m s^{-2}
β	β	2×10^{-11} m^{-1} s^{-1}
A	Horizontal viscosity	1×10^{4} m^{2} s^{-1}
Case 2		
(Lx, Ly)	Model region	20 000 km × 8000 km
$\Delta x, \Delta y$	Grid resolution	67 km, 67 km
Δt	Time step	1570 s
g'	Reduced gravity	0.049 m s^{-2}
β	β	2×10^{-11} m^{-1} s^{-1}
A	Horizontal viscosity	1×10^{3} m^{2} s^{-1}

might provide an efficient representation of this data. In other words, one can obtain an ''efficient'' representation when the basis elements are ''similar'' to the features in a given flow. This similarity can be made precise by simply counting the dimension of the space that one uses with a given error tolerance.

Now let us consider Daubechies-based wavelet Galerkin methods. Daubechies wavelets have the remarkable property that they are compactly supported in the physical space and orthogonal under translation and dilation. Assuming that one's initial condition is not a basis function, that is, a wavelet, then one must employ a quadrature formula. This quadrature formula provides the link between physical space point values and the wavelet subspace at the finest scale. When one observes the action of an adaptive Daubechies wavelet-based numerical method on these physical space point values, one finds an adaptive grid finite-difference method with an order of accuracy roughly double that of the approximation accuracy for the wavelet at hand, that is, superconvergence (see Jameson 1993a,b).

It is possible to make a few general statements concerning wavelet Galerkin methods. Let Galerkin methods be summarized as methods in which the degrees of freedom are the expansion coefficients of a set of basis functions. These expansion coefficients are, by definition, not in the physical space. However, partial differential equations are generally specified as equations, with boundary conditions, in the physical space, so that nonlinearities, etc. when treated in a wavelet subspace are often unnecessarily complicated. In short, if a quantity is specified in a given space, then implementation is most straightforward in that space. There appears to be no compelling reason to work with Galerkin-style coefficients in a wavelet method, and to try to create a practical Galerkin-style method with wavelets seems to be unnecessarily difficult when an equivalent method can be implemented in the physical space.

Boundary conditions that are not periodic are one of the weakest points of wavelet methods. For wavelet Galerkin methods it is unlikely that sufficient accuracy will ever be achieved at the boundary (see Jameson 1996a). Beyond sufficient accuracy, it is far from a straightforward task to impose even the simplest, nonperiodic, boundary conditions for wavelet Galerkin methods.

A second obstacle with Galerkin methods is the evaluation of nonlinear terms. That is, if one has the wavelet coefficients of $u(x)$, then how can one obtain easily the coefficients of $u^2(x)$? Furthermore, what if the nonlinearity is $e^{u(x)}$? In short, one need not struggle with all the problems of implementing a Galerkin method if one chooses to work in the physical space from the beginning.

In contrast to Galerkin-based numerical methods, collocation methods involve numerical operators acting on point values in the physical space. Generally, wavelet collocation methods are created by choosing a wavelet and some kind of computational grid structure, which will be dynamically adapted. Recall, the approximation properties of wavelet methods are such that they are constructed in order to reproduce algebraic polynomials perfectly up to a given order. Also recall, finite-difference methods are constructed from an underlying algebraic polynomial. When one adds the adaptivity of a ''wavelet'' collocation method, one obtains in the physical space operations that are exact for some set of polynomials and performed on a nonuniform grid in the

physical space. In effect, one obtains finite differencing on nonuniform grids in the physical space. In addition, the operators are far from optimal in obtaining a given accuracy for a given stencil width. The stencil is usually longer than is needed, with the extra degrees of freedom having no specific function.

3. A short review of wavelet analysis

To define Daubechies-based wavelets, see Daubechies (1988) for the original work, consider the two functions $\phi(x)$, the scaling function, and $\psi(x)$, the wavelet. The scaling function is the solution of the dilation equation,

$$\phi(x) = \sqrt{2} \sum_{k=0}^{L-1} h_k \phi(2x - k), \tag{1}$$

where $\phi(x)$ is normalized $\int_{-\infty}^{\infty} \phi(x)\, dx = 1$, and the wavelet $\psi(x)$ is defined in terms of the scaling function,

$$\psi(x) = \sqrt{2} \sum_{k=0}^{L-1} g_k \phi(2x - k). \tag{2}$$

One builds an orthonormal basis from $\phi(x)$ and $\psi(x)$ by dilating and translating to get the following functions:

$$\phi_k^j(x) = 2^{-j/2} \phi(2^{-j}x - k), \quad \text{and} \tag{3}$$

$$\psi_k^j(x) = 2^{-j/2} \psi(2^{-j}x - k), \tag{4}$$

where $j, k \in Z$; j is the dilation parameter; and k is the translation parameter. The coefficients $H = \{h_k\}_{k=0}^{L-1}$ and $G = \{g_k\}_{k=0}^{L-1}$ are related by $g_k = (-1)^k h_{L-k}$ for $k = 0, \ldots, L-1$. All wavelet properties are specified through the parameters H and G. Here $\psi(x)$ has M vanishing moments, which determines the accuracy. In other words, $\psi_k^j(x)$ will satisfy

$$\delta_{kl}\delta_{jm} = \int_{-\infty}^{\infty} \psi_k^j(x) \psi_l^m(x)\, dx, \tag{5}$$

where δ_{kl} is the Kronecker delta function, and the accuracy is specified by requiring that $\psi(x) = \psi_0^0(x)$ satisfy

$$\int_{-\infty}^{\infty} \psi(x) x^m\, dx = 0, \tag{6}$$

for $m = 0, \ldots, M-1$.

For Daubechies wavelets the number of coefficients in H and G, or the length of the filters H and G, denoted by L, is related to the number of vanishing moments M by $2M = L$.

As is expected, the regularity increases with the support of the wavelet. The usual notation to denote a Daubechies-based wavelet defined by coefficients H of length L is D_L.

It is usual to let the spaces spanned by $\phi_k^j(x)$ and $\psi_k^j(x)$ over the parameter k, with j fixed, be denoted by V_j and W_j, respectively,

$$V_j = \operatorname*{span}_{k \in Z} \phi_k^j(x), \tag{7}$$

$$W_j = \operatorname*{span}_{k \in Z} \psi_k^j(x). \tag{8}$$

The spaces V_j and W_j are related by

$$\cdots \subset V_1 \subset V_0 \subset V_{-1} \subset \cdots, \quad \text{and} \tag{9}$$

$$V_j = V_{j+1} \oplus W_{j+1}, \tag{10}$$

where the notation $V_0 = V_1 \oplus W_1$ indicates that the vectors in V_1 are orthogonal to the vectors in W_1 and the space V_0 is simply decomposed into these two component subspaces.

4. The wavelet-optimized finite-difference method

Explanations of the one-dimensional version of WOFD have appeared in Jameson (1993a), Jameson (1994), and Erlebacher et al. (1996). The first occurrence of the argument that wavelets should be used only to analyze computational data for error detection and grid generation occurred in Jameson (1993a) and subsequently in Jameson (1994). Let us motivate the argument supporting WOFD by addressing a number of relevant points with respect to wavelet numerical methods.

a. One-dimensional WOFD

We begin by reviewing the one-dimensional version of WOFD. In short, we can say that the key distinguishing feature of this version of WOFD is that wavelets are proposed a method for grid refinement.

The idea of using wavelets to generate numerical grids began with the observation in Jameson (1993b) that the essence of an adaptive wavelet-Galerkin method is nothing more than a finite-difference method with grid refinement. So, instead of letting the magnitude of wavelet coefficients choose which basis functions to use in a Galerkin approach, let the same coefficients choose which grid points to use and then think of the wavelet method in a collocation sense.

In other words, suppose a calculation begins with N evenly spaced samples of a function $\mathbf{f}$ and that some quadrature method produces N scaling function coefficients on the finest scale denoted by V_0. If the spacing between adjacent values in the vector $\mathbf{f}$ is Δx, then this is also the physical-space resolution of any calculation done in V_0. Now, decompose V_0 once to get $V_0 = V_1 \oplus W_1$. Similarly speaking, the physical space resolution of V_1 is $2\Delta x$ and the refinement from the $2\Delta x$ physical-space resolution to the Δx physical-space resolution is dictated by the wavelet coefficients in W_1.

Given a vector of evenly spaced data $\mathbf{s}$, which will be considered the scaling function coefficients on the finest scale, find the scaling function and wavelet coefficients on the next coarsest scale:

$$s_k^j = \sum_{n=1}^{2M} h_n s_{n+2k-2}^{j-1} \quad \text{and} \tag{11}$$

$$d_k^j = \sum_{n=1}^{2M} g_n s_{n+2k-2}^{j-1} \tag{12}$$

(see above for notation definitions). One can continue this type of decomposition in order to obtain wavelet coefficients on a number of scales, $\mathbf{d}_1$, $\mathbf{d}_2$, ... This is all the information that is necessary in order to choose a numerical grid. That is, if d_k^j is the wavelet coefficient at scale j and location k, then a grid point, or two, is added at location k and scale j. For example, if coefficient $|d_5^2| > \epsilon$, where ϵ is a user-defined sensitivity or wavelet threshold, then one can add a grid point at location x_{20}, since the wavelet coefficients at scale $j = 2$ represent local high frequencies in the physical space at scale $4\Delta x$, that is, $5 \times 4\Delta x = x_{20}$. Here x_i represents the numerical value of the ith grid point. Note that one can add a number of grid points in any region around large wavelet coefficients, and it is generally more efficient to do so. In addition, if one is calculating a moving wave structure, then the grid points can be added in front of the wave structure motion. The wave velocity can easily be estimated from the information obtained from wavelet coefficients at two different times. Note that a Fortran software implementation of this grid refinement algorithm can be found in Jameson (1996b).

Next, we must choose an appropriate wavelet. We will restrict ourselves to the orthogonal class of wavelets that we discussed above and which are known as Daubechies wavelets. Perhaps restricting ourselves to only orthogonal wavelets is perhaps not necessary, but the logic behind WOFD stems directly from the Daubechies class of wavelets and hence we choose to stay within the Daubechies realm. The Daubechies wavelets are commonly labeled D_2, D_4, . . . , D_{2M}, where the subscript $2M$ indicates twice the number of vanishing moments of the wavelet, which is also equal to the number of nonzero wavelet filter coefficients previously denoted as h_k. There are two paths that one can follow in choosing which wavelet filter to use. One is to follow the logic of WOFD and the second is to choose a wavelet filter based on practical application and experience, in either case we arrive at D_4 as the best wavelet for our application. From the point of view of a practical filter one can observe (see Daubechies 1988) that the wavelets and also wavelet filter coefficients for D_6 and above are quite long filters with the tail of the filter close to but not equal to zero. The portion of the filter that is near zero has no practical impact on the filtering capabilities and furthermore on a computer one generally tries to avoid numbers that are close to but not equal to zero since roundoff error can obscure the two. The shortest filter, D_2, on the other hand, is simply too short to catch relevant features in practice. From the point of view of WOFD logic, we want the order of the wavelet superconvergence to be equal to the order of the numerical scheme (see Jameson 1993b).

Our discussion has so far been limited to spatial discretization, but certainly we must comment on time advancement. If possible one's time advancement should be the same order as the spacial discretization. Sometimes this is not possible, say, in the case of spectral methods. But, for spacial schemes of order less than 4, it is best to keep the time advancement on the same order. In our case for WOFD we are using a fourth-order spacial discretization and so we choose the fourth-order time advancement of Runge–Kutta.

One should note that the grid refinement mechanism defined above can be applied to a variety of numerical methods. In fact, one needs only to find a quantity that can be interpolated such that it occurs at evenly spaced intervals. Consider, for example, schemes that operate on cell-averaged quantities for conservation laws. In this case the values that one would perform the wavelet analysis on would be the cell averages and the result of the wavelet analysis would be the selection of appropriate cell or volume sizes. In fact, this approach has been applied to the numerical scheme known as the essentially non-oscillatory (ENO) method (see Erlebacher et al. 1996).

Errors of numerical origin such as reflection and aliasing at interfaces between different grid densities for WOFD are quite small and will be on the order of the wavelet threshold. The fundamental reason that WOFD reflection is small is that the polynomial interpolation is in the middle of the polynomial, that is, central differencing, instead of at the ends of the polynomial, as is done in methods with boundaries. For example, in methods where a subdomain with a fine grid is placed into a larger domain, there is clearly a new boundary introduced into the computational area. At the edge of each subdomain one will execute some kind of one-sided differencing. With WOFD, on the other hand, the fourth-order, five-point numerical stencil extends smoothly across interfaces between differing gridpoint densities, and all differencing is executed in the center of the stencil keeping artifacts to a minimum.

b. The two-dimensional WOFD

This section will explain the two-dimensional version of WOFD. As stated above, the one-dimensional version of WOFD has appeared in other publications, but this is the first appearance and explanation of the two-dimensional version of the scheme.

1) MINIMIZING WORK AND STORAGE

First, we should state the ultimate goals behind any adaptive scheme. Primarily we want to minimize two quantities, the number of floating point operations and the amount of computer memory that is needed during the computation. By keeping these two quantities to a

minimum then it is possible to run much larger computer models on a given generation of computers by doing computational work only where computational work is needed. In two dimensions, minimizing these two quantities simultaneously requires far more effort than in one dimension.

Minimizing the number of floating point operations is, if you will, the easy part since it can be traced directly to how many grid points are used in the calculation. Storage minimization, on the other hand, requires careful attention to data structures. WOFD dimensions all variables as long one-dimensional arrays with pointers to the locations of the data that are needed at a given time during the calculation. To illustrate, if $h(K)$ represents, say, sea surface height in a one-dimensional array, then a pointer $K(i, j)$ is used to give the integer pointing to the location where the Cartesian coordinate value i, j of h is stored. We will discuss WOFD pointers a bit more later. In this manner the storage is kept to a fraction of the finest possible grid. The next subsection will give an overview of how the grid points are selected.

2) WAVELET-BASED GRID SELECTION

Detecting structures in two dimensions is far more complex than detecting them on a line as in one dimension. It is critical to ensure that the grid selection mechanism will not miss anything in the domain such as structures aligned along an axis or objects moving along or orthogonal to an axis.

The two-dimensional grid selection mechanism used here is fundamentally the one-dimensional wavelet grid selection mechanism explained above applied in a tensor product manner. A number of steps are, however, required since at any time during the calculation the data exists on a nonuniform grid, whereas wavelet analysis requires uniform grid data. The grid selection steps are as follows:

- Along each row and column of the data, one interpolates the nonuniform grid information onto a uniform grid for each physical variable.
- On each of these uniform grid sets of one-dimensional information the previously explained one-dimensional wavelet grid selection mechanism is used for each physical variable.
- If a given grid point is selected from either a row or a column, then the numerical value for this grid is stored for each physical variable in a one-dimensional array of data as explained above and the appropriate pointers are set for proper bookkeeping.

The above grid selection mechanism is complex for a few reasons, one of which is the need to keep the storage down to a minimum as we discussed above. In the above two-dimensional grid selection scheme, the physical quantities are never on a full two-dimensional uniform grid, but are only defined on one-dimensional uniform grids and only at the time that a new grid is selected.

Once the grid is chosen one must build difference operators.

3) DETERMINING FIXED GRIDS AND MOVING GRIDS

The grid generation mechanism that we have outlined in this paper is designed specifically for flows where one might need a great deal of grid movement in order to accomplish an efficient calculation. As our example shows, we can follow structures as they move from one side of the domain to the other side while keeping most of the grid points around this structure. We consider the moving grid problem to be far more challenging than, say, determining an appropriate gridpoint structure that will remain fixed during a calculation. In oceanography often one is satisfied with a nonuniform grid that does not change with time, but how does one determine the appropriate nonuniform grid? If one simply places more grid points where there is more activity without some scientific measure of computational errors, then one certainly cannot obtain an optimal gridpoint structure. We can, therefore, recommend that the WOFD grid selection mechanism be used to determine appropriate grids that will not change with time. One can do this by, say, analyzing the initial condition of the flow and perhaps by periodically turning on the WOFD grid selection mechanism to confirm that during the calculation that the fixed grid is in fact appropriate. However, we restate that while determining an appropriate fixed grid is one application of WOFD, it is not considered the primary challenge. The primary challenge is to have a completely moving grid structure that is a function of time.

4) POINTERS

The two-dimensional version of WOFD presented in this paper was designed for a single serial processor. In this scenario one can keep storage to a minimum by packing all the data in a long one-dimensional array and pointing to the data for retrieval as it is needed.

First, in order to make the loop structure efficient within the software, two integer arrays are created that store the number of grid points along each column and each row. For example, the integer array Nkeepx(i) is created such that in row i one will find Nkeepx(i) grid points. Likewise, the array Nkeepy(j) is constructed such that in column j one will find Nkeepy(j) grid points.

In conjunction with these two integer arrays, we construct two pointer arrays fx(i, j) and fy(i, j), which keep track of which grid points are actually used. For example, in row i one finds Nkeepx(i) grid points and these grid points are pointed to by fx(i, j) where the index j ranges from 0 to Nkeepx(i). In this manner the loops in the serial code execute efficiently and the storage is kept to a minimum.

5) Higher-order schemes and WOFD

Let us define high-order numerical schemes as schemes that are of order four or higher. This definition is fairly arbitrary, but conforms to the language that is currently used in the numerical analysis community. Furthermore, it also seems appropriate for the oceanography community since it appears that second-order schemes are the norm.

A higher-order numerical scheme simply means that the numerical approximation is done with a high-order polynomial. In other words, all numerical schemes can be seen as originating from polynomial interpolation: a high-order scheme uses a high-order polynomial and a low-order scheme uses a low-order polynomial. This also holds for spectral methods where the polynomials are of maximum order and are of either the algebraic type for Chebyshev spectral methods or trigonometric type for Fourier spectral methods.

The key question is when is a high-order scheme appropriate. This question is addressed in Jameson (1998). To summarize the results presented in this paper, it can be said that one chooses the order of the scheme such that the polynomial interpolation of the numerical data has as little error as possible. Therefore, if the data is very rough and localized then a low-order scheme with a high gridpoint density is appropriate. On the other hand, if the numerical data is composed of large-scale smooth features, then one should use high-order polynomials on a coarse grid. The test of which order is best is conducted with wavelet analysis.

Another issue that should influence the choice of the order of the scheme is the length of the numerical integration. Simply said, all numerical schemes have errors and lower-order schemes have larger errors than high-order schemes when applied to smooth data. Under the assumption that the data is ''smooth,'' low-order numerical schemes will produce an error at each time step. If one performs a ''long-time'' integration, perhaps a few thousand or more time steps, then the accumulated error from a low-order scheme can completely corrupt the physics. Whereas, a higher-order scheme will produce less error at each time step and can produce numerical results that are much closer to the actual physics. For discussions in oceanographic context, see Sanderson (1998).

In general one can build a numerical scheme of any order on any grid, uniform or not. In the next section we will discuss the explicit construction of such general numerical schemes.

6) Building the difference operators

In general one can interpolate with a variety of functions such as trigonometric polynomials, Gaussians, etc. (see Jameson 1998). Here we will explore the traditional manner of using algebraic polynomials.

Interpolation with algebraic polynomials is probably the most common form of interpolation, and it is from this type of interpolation that common uniform grid finite-difference methods can be found. Using the following formula one can find the finite-difference coefficients for an arbitrary grid and of arbitrary order. One simply fits the polynomial to the data, followed by differentiation of the polynomial, and finally one evaluates the polynomial at the point of interest. The well-known Lagrangian interpolation formula for algebraic interpolation is

$$A_j(x) = \prod_{k=0,k\neq j}^{n} (x - x_k) \Big/ \prod_{k=0,k\neq j}^{n} (x_j - x_k), \quad (13)$$

where $A_j(x_k) = \delta_{jk}$. For given values $w_0, w_1, \ldots, w_n$, the polynomial

$$p_n(x) = \sum_{k=0}^{n} w_k A_k(x), \quad (14)$$

is in P_n and takes on these values at the points x_i:

$$p_n(x_k) = w_k, \quad (15)$$

for $k = 0, 1, \ldots, n$, and it is the truncation error of this interpolation that determines the order of accuracy that one obtains when differentiating.

Note that all nonspectral numerical methods begin with a low-order local algebraic polynomial approximation to the numerical data. To be a bit more concrete, in the fourth-order case, a fourth-order polynomial is fit to five neighboring points. Note that there is no need for these points to be uniform. A fourth-order polynomial that has five free parameters to choose can be passed uniquely through five points regardless of whether the points are uniform or not. Now, to use this fourth polynomial in order to find a fourth-order derivative simply involves taking a derivative of this polynomial and choosing a point at which to evaluate it. To have central derivative without bias in direction, WOFD uses the central point. In the following equations we give the formulas for the central point derivative and the two points near the boundary, which are needed when the boundary conditions are not periodic.

The following five coefficients,

$$\begin{aligned}
c(1) &= [(-x2 + x3)/(x1 - x2)][(x3 - x4)/(x1 - x3)] \\
&\quad \times [(x3 - x5)/(x1 - x4)][1.0/(x1 - x5)], \\
c(2) &= [(-x1 + x3)/(-x1 + x2)][(x3 - x4)/(x2 - x3)] \\
&\quad \times [(x3 - x5)/(x2 - x4)][1.0/(x2 - x5)], \\
c(3) &= 1.0/(-x1 + x3) + 1.0/(-x2 + x3) \\
&\quad + 1.0/(x3 - x4) + 1.0/(x3 - x5), \\
c(4) &= [(-x1 + x3)/(-x1 + x4)] \\
&\quad \times [(-x2 + x3)/(-x2 + x4)] \\
&\quad \times [(x3 - x5)/(-x3 + x4)][1.0/(x4 - x5)], \\
c(5) &= [(-x1 + x3)/(-x1 + x5)] \\
&\quad \times [(-x2 + x3)/(-x2 + x5)] \\
&\quad \times [(x3 - x4)/(-x3 + x5)][1.0/(-x4 - x5)],
\end{aligned}$$

are the five coefficients that would be used to find a fourth-order central difference on an arbitrary grid; that is,

$$f'(i) = \sum_{j=1}^{j=5} c(j) f(i + j - 3). \tag{16}$$

The variables $x1, \ldots, x5$ represent the five grid point values.

If the boundary conditions are not periodic then we use one-sided derivatives at the left- and right-hand boundaries. Here we provide the one-sided derivatives for the left side of the domain. Beginning with the left-most point we have

$$\begin{aligned}
c_L(1) &= 1.0/(x1 - x2) + 1.0/(x1 - x3) \\
&\quad + 1.0/(x1 - x4) + 1.0/(x1 - x5), \\
c_L(2) &= [(x1 - x3)/(-x1 + x2)][(x1 - x4)/(x2 - x3)] \\
&\quad \times [(x1 - x5)/(x2 - x4)][1.0/(x2 - x5)], \\
c_L(3) &= [(x1 - x2)/(-x1 + x3)][(x1 - x4)/(-x2 + x3)] \\
&\quad \times [(x1 - x5)/(x3 - x4)][1.0/(x3 - x5)], \\
c_L(4) &= [(x1 - x2)/(-x1 + x4)][(x1 - x3)/(-x2 + x4)] \\
&\quad \times [(x1 - x5)/(-x3 + x4)][1.0/(x4 - x5)], \\
c_L(5) &= [(x1 - x2)/(-x1 + x5)][(x1 - x3)/(-x2 + x5)] \\
&\quad \times [(x1 - x4)/(-x3 + x5)][1.0/(-x4 + x5)],
\end{aligned}$$

where the derivative would be found as

$$f'(1) = \sum_{j=1}^{5} c_L(j) f(j). \tag{17}$$

To find the derivative at the grid point that is second from the left-hand side, the coefficients would be

$$\begin{aligned}
c_{L+1}(1) &= [(x2 - x3)/(x1 - x2)][(x2 - x4)/(x1 - x3)] \\
&\quad \times [(x2 - x5)/(x1 - x4)][1.0/(x1 - x5)], \\
c_{L+1}(2) &= 1.0/(-x1 + x2) + 1.0/(x2 - x3) \\
&\quad + 1.0/(x2 - x4) + 1.0/(x2 - x5), \\
c_{L+1}(3) &= [(-x1 + x2)/(-x1 + x3)] \\
&\quad \times [(x2 - x4)/(-x2 + x3)] \\
&\quad \times [(x2 - x5)/(x3 - x4)][1.0/(x3 - x5)], \\
c_{L+1}(4) &= [(-x1 + x2)/(-x1 + x4)] \\
&\quad \times [(x2 - x3)/(-x2 + x4)] \\
&\quad \times [(x2 - x5)/(-x3 + x4)][1.0/(x4 - x5)], \\
c_{L+1}(5) &= [(-x1 + x2)/(-x1 + x5)] \\
&\quad \times [(x2 - x3)/(-x2 + x5)] \\
&\quad \times [(x2 - x4)/(-x3 + x5)][1.0/(-x4 + x5)],
\end{aligned}$$

where the derivative would be found as

$$f'(2) = \sum_{j=1}^{5} c_L(j) f(j). \tag{18}$$

7) BOUNDARY CONDITIONS

One of the key problem areas for most wavelet methods is the difficulty in implementing general boundary conditions. If one carefully examines the literature they will find a large number of wavelet methods are either implemented only with periodic boundary conditions or they change over to polynomial interpolation at the boundaries. In other words, great effort has been made to work within the strict theoretical framework of wavelet analysis, but if one switches over to polynomial interpolation at the boundaries then all of the strict theoretical framework has no meaning. WOFD, on the other hand, is physical space implementation, and all boundary conditions can be implemented directly using existing finite-difference technology and methods. This a key point that overcomes a severe deficiency of many other methods. For the example problem given in a later section we have periodic boundary conditions in the direction of primary wave motion, but in the direction orthogonal to this motion we have a no-slip boundary condition. For the fourth-order scheme this entails one-sided difference operators as given above. With these one-sided difference operators one can implement the boundary condition directly at the outermost grid point.

8) COST AND BENEFIT OF WOFD

Of course, all adaptive grid methods have some cost associated with the adaptation mechanism and other overhead that one does not encounter with uniform grid techniques. Let us examine the cost in terms of floating point operations (FLOPS; we will only count multiplies for convenience). Note that in order to measure the effectiveness of the WOFD we always compare the reduced grid calculation to the calculation performed without grid refinement. One desires to obtain roughly the same answer as the uniform grid but executed on the less costly nonuniform grid. This gives a precise way to compare the work savings and to observe how much error has been introduced by using a reduced grid.

The FLOPS of simply taking the wavelet transform depend on the wavelet chosen. As described above, we are using the D_4 wavelet, which is a filter of four nonzero numbers. The number of FLOPS for taking one wavelet decomposition, that is, going from V_0 to $V_1 \oplus W_1$ is simply $4N$ where N is the total number of grid points on the finest uniform grid. To execute a second wavelet decomposition that would take us from V_1 to $V_2 \oplus W_2$ requires $4N/2$ FLOPS. Generally, in a practical setting one executes around three or four wavelet decompositions. One can see that the maximum number of FLOPS will at most be $4(2N)$. In fact, this is roughly twice the cost of taking a single derivative on a uniform fine grid.

And, if one considers that the time advancement of the fourth-order Runge–Kutta scheme requires four derivatives to be taken, then it is clear that the wavelet decomposition costs less than one time step on the finest grid. Generally there is a bit more cost associated with the grid refinement mechanism such as interpolating onto a fine grid from a coarse grid and building the differentiation matrices. However, the total cost of all the grid refinement overhead remains less than two time steps on a finest uniform grid. Of course, one does not change the grid at every time step, but perhaps at every 10th or 100th or 1000th time step, depending on the characteristics of the flow; that is, a slowly varying flow might require a new grid only every 1000th time step. The less often one updates, the more cost effective the WOFD technique becomes.

To measure the savings produced by using this adaptive technique, one needs only to count the number of grid points used during the calculation. For example, in the problem presented later in this paper we generally used, say, one-quarter of the grid points that would be available on the fine grid. By doing so, we have generally reduced the work by a factor of four. And, if one does not refine often, then this gridpoint fraction becomes the dominate measure of work involved in terms of FLOPS. The benefit is, as stated above, that one can obtain the uniform grid solution with far less work.

5. Example in oceanography

We choose equatorial Kelvin and Rossby wave propagation initiated by imposed anomaly of the upper-layer thickness as an example.

This model is a one-layer reduced gravity model (a so-called 1.5-layer model). A top active layer of constant density ρ and variable thickness h overlays a layer of infinite depth and constant density $\rho + \Delta\rho$ in which there is no motion.

The model equations in Cartesian coordinates are

$$\frac{\partial uh}{\partial t} + \frac{\partial uuh}{\partial x} + \frac{\partial uvh}{\partial y} - fhv = -g'h\frac{\partial h}{\partial x} + A\nabla^2 uh, \quad (19)$$

$$\frac{\partial vh}{\partial t} + \frac{\partial uvh}{\partial x} + \frac{\partial vvh}{\partial y} + fhu = -g'h\frac{\partial h}{\partial y} + A\nabla^2 vh, \quad (20)$$

$$\frac{\partial h}{\partial t} = -\left(\frac{\partial hu}{\partial x} + \frac{\partial hv}{\partial y}\right), \quad (21)$$

where u and v are velocities of x direction and y direction, respectively; f is Coriolis parameter; g' is reduced gravity ($g\Delta\rho/\rho$); A is horizontal mixing coefficient. Equations (19) and (20) are momentum equations and (21) is a continuity equation.

The parameter values used here are summarized in Table 1. The initial conditions are zero velocity and a Gaussian bell-shaped perturbation in the upper-layer thickness, symmetrical about the equator

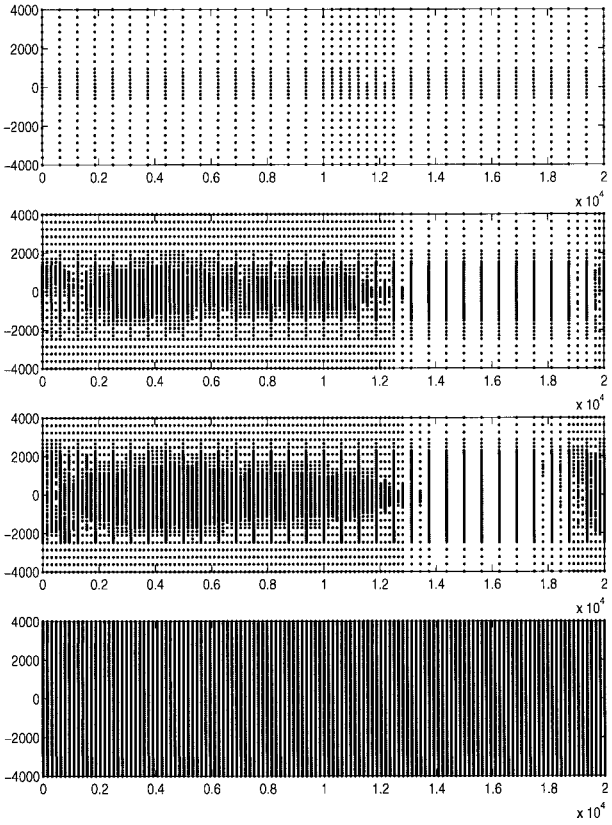

FIG. 4. Four grids for four different flows at day 35.

$$h = H + \delta h \exp\left[-\left(\frac{x^2}{l_x^2} + \frac{y^2}{l_y^2}\right)\right], \quad (22)$$

where 40 and 60 m are chosen as H and δh, respectively; $l_x = 667$ km, and $l_y = 334$ km. The gravity wave speed C is $\sqrt{g'H} = 1.96$ m s^{-1}, so that the equatorial radius of deformation $\sqrt{C/\beta}$ is 313 km. The finite difference is fourth-order in space and time.

Figures 1, 3, and 5 show the time sequence of upper-layer thickness in case 1 (Table 1). The initial perturbation is split into an eastward propagating Kelvin wave and westward Rossby waves. The solution compares favorably with a well-known numerical solution (e.g., Philander et al. 1984). In Fig. 1, we see the initial state of the variable h and our four different wavelet thresholds. The top plot is for the wavelet threshold of 0.001 (corresponding to error of ~1 m of h). The plot second from the top is for the wavelet threshold of 0.0001 (~0.1 m). The plot third from the top is for the threshold of 0.00001 (~0.01 m), and the bottom plot is the uniform grid solution, which is equivalent to setting the wavelet threshold to 0. In Fig. 2, we see the wavelet generated grids associated with these four wavelet thresholds. Note that there are far fewer grid points in the top figure yielding a great reduction in computational work. In Fig. 3, we see the four flows at day 35 and the four corresponding grids in Fig. 4. Note how the bottom threes

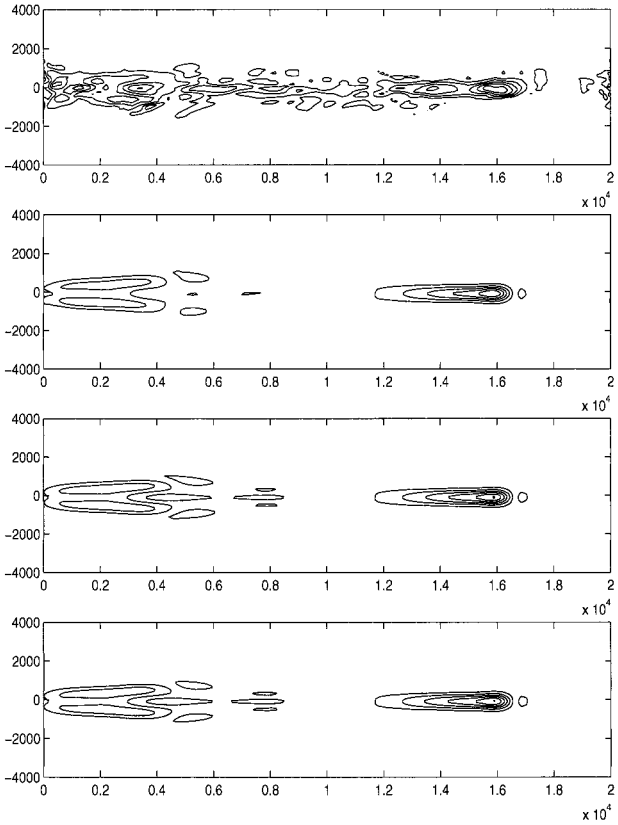

FIG. 5. Four flows at final time, day 70. Contour interval is 2 m.

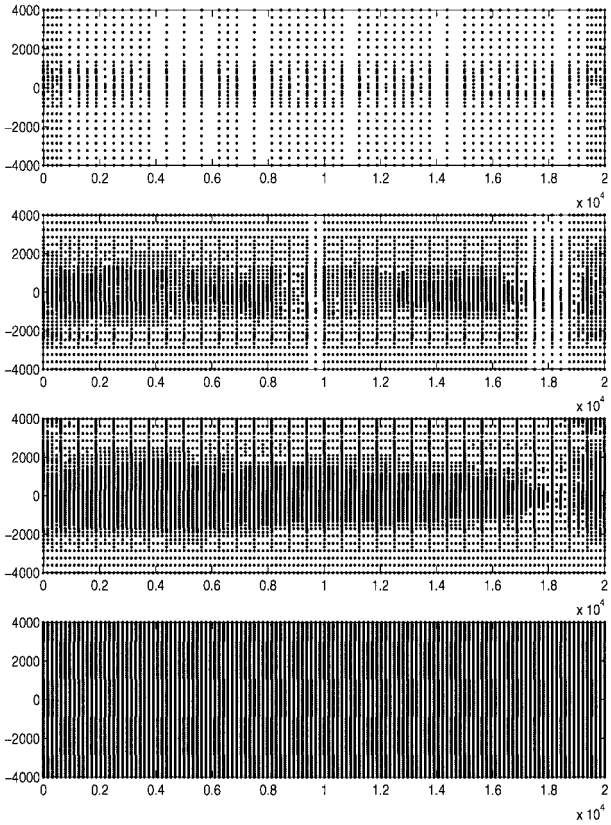

FIG. 6. Four wavelet grids at final time.

flows visually are very similar whereas the top flow appears to have a large error. In Fig. 5, we see the four flows at day 70 and the corresponding wavelet generated grids in Fig. 6. Again, note that the bottom three flows are very similar visually whereas the top flow seems to have broken up a great deal. But, even with the poor solution obtained in the top flow, one can note that the main structures are in about the same place as in the bottom three flows. As a user, one must always choose between computational work and the quality of the solution. Out of the four possible wavelet thresholds that we have presented here, the optimal choice would most likely be the flow second from the top that corresponds to the wavelet threshold set to 0.0001. In this flow we obtain a good qualitative solution while keeping the computational work down to a minimum.

As seen in Boyd's (1998) application to Kelvin wave frontogenesis, high-order scheme is advantageous to reduce the numerical errors for a modest expenditure of computational cost. Figure 7 shows the upper-layer thickness and corresponding grids of case 2 (Table 1) with wavelet threshold 0.0001. In Fig. 7, we see the benefits from the fourth-order space and time scheme: the very fine structure of waves (e.g., sharp Kelvin wave front, a group of westward propagating short waves). The detail of the physics in this diagram cannot be obtained with a lower-order scheme of similar grid size (e.g., see Ginis et al. 1988, their Fig. 8b). In Fig. 8, we see the same simulation at a later time in the simulation.

6. Errors as a function of the wavelet threshold

For the purpose of testing numerical schemes, it is ideal to know the exact solution of partial differential equations at hand. However, as the partial differential equations become more complicated, it is unlikely that such an exact solution can be found. Therefore for adaptive schemes it is informative to separate the affect of the adaptation process from the uniform grid solution. In other words, the uniform grid solution at a given gridpoint density will be compared to adaptive grid solutions where the finest grid point density matches the uniform grid solution.

Generally numerical approximations of nonlinear physics will have L_∞ errors that are much larger than the errors comitted by adapting. In other words, WOFD errors that can be attributed to the adapting process are generally on the order of the wavelet threshold used and common wavelet thresholds are 10^{-3} or perhaps 10^{-5}, whereas errors committed by numerically approximating the nonlinear physics will be on the order of, say, 0.1 or 0.01. Such error estimates can rarely be verified.

For the numerical simulations presented in section 5, we have verified the relationship between the wavelet

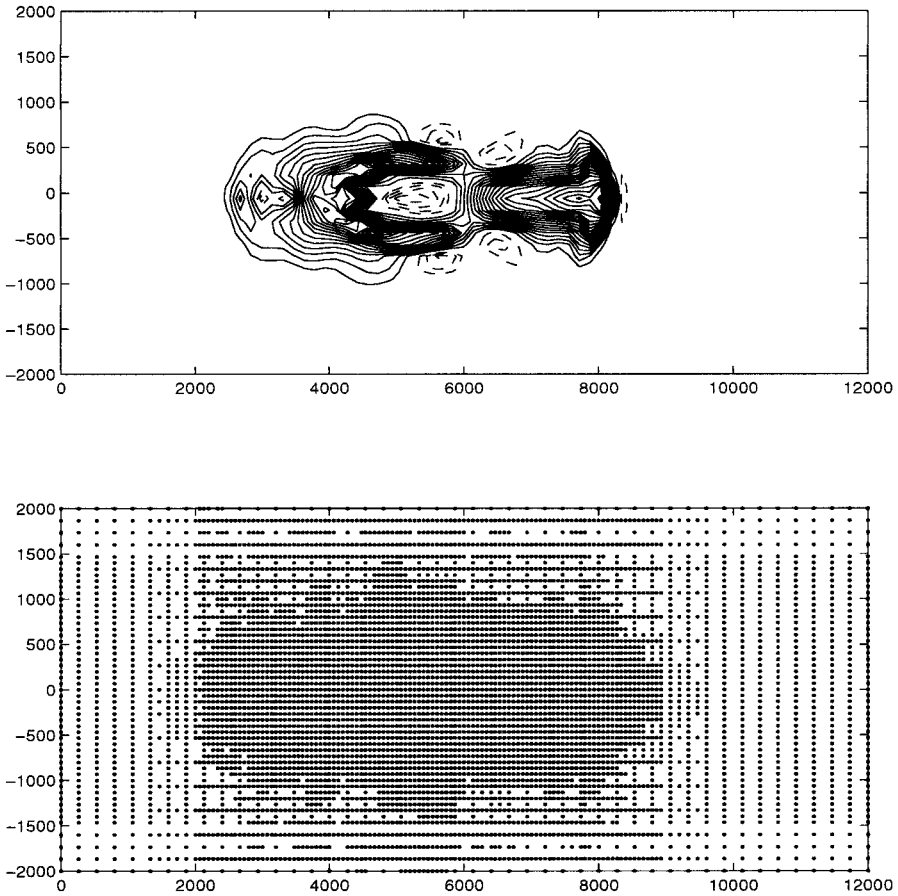

FIG. 7. Rossby and 7 Kelvin waves in case 2 at day 15. Variable h is shown. Contour interval is 1 m. Dashed line is negative anomaly.

threshold and the L_∞ difference between the uniform grid solution and the adaptive grid solution. In Fig. 9, we see the L_∞, or simply maximum, difference between compressed calculation and the uniform grid, or uncompressed, calculation shown in Figs. 1, 3, and 5. We will call this difference the error due to compression. We can see from the plot that there are three different error curves corresponding to three simulations run at three different wavelet thresholds in case 1. The top curve is for the wavelet threshold set to 0.001 (corresponding to $h \sim 1$ m), the middle curve for the wavelet threshold set to 0.0001 ($h \sim 0.1$ m), and bottom curve is for the threshold set to 0.00001 ($h \sim 0.01$ m). We can see that all three curves tend toward a plateau value that is slightly larger than the specified wavelet threshold. In other words, the user can specify the maximum error that can be tolerated during the simulation and set the wavelet threshold accordingly. This gives the user a great deal of control over computational error while minimizing the computational cost.

Again, the wavelet threshold will determine roughly the difference between the adaptive solution and the uniform grid solution. But, generally speaking, the errors committed on the fine grid will be orders of magnitude larger than the additional errors added by adapting.

7. Conclusions

The purpose of the work has been to introduce the two-dimensional version of the Wavelet-Optimized Finite Difference Adaptive High Order (WOFD-AHO) numerical method and to introduce wavelet adaptive grid methods to the oceanographic community. Generally speaking, adaptive numerical methods, if done properly, can provide computational solutions that greatly excede nonadaptive numerical method solutions for a given generation of computers. So, we believe that for those who are interested in obtaining the best numerical solution at any given time that adaptive numerical methods are necessary. All adaptive numerical methods incorporate some kind of adaptive mechanism that is often based on some physical properties in the flow such as steep gradients in a given flow field. Furthermore, these adaptive mechanisms are usually tailored specifically for the given flow and are often somewhat ''tweaked,'' which makes it difficult for a similar numerical experiment to be repeated by a different researcher. Wavelets,

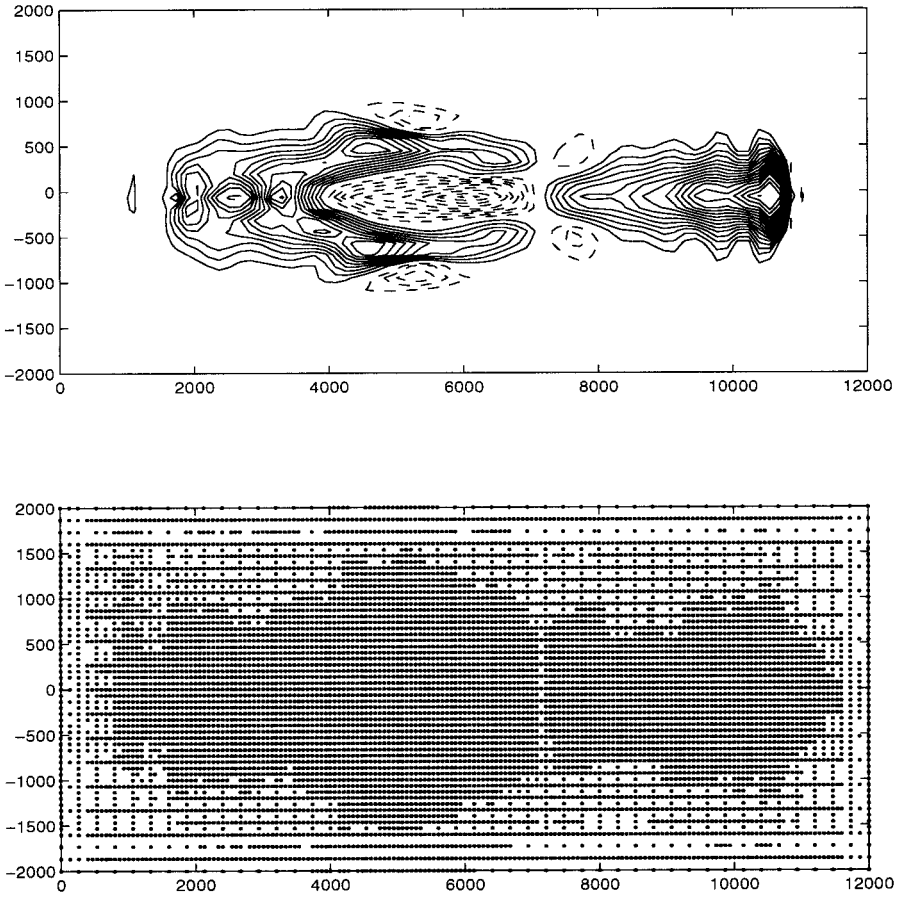

FIG. 8. Same as Fig. 7 except for at day 30.

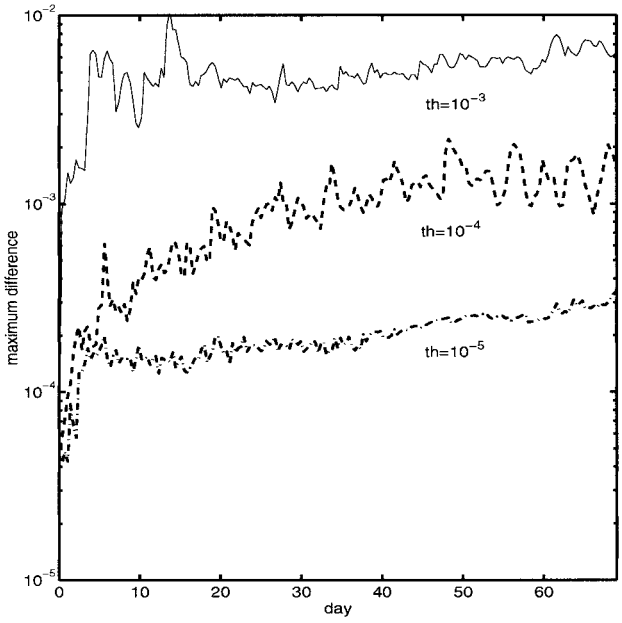

FIG. 9. The time series of maximum layer thickness h difference from the solution using the finest grid density. The difference 10^{-3} corresponds to 1 m in physics sense.

on the other hand, provide a grid refinement and coarsening mechanism that does not require tweaking or any special adaptation to a given flow field or numerical experiment. We believe that we have shown that the two-dimensional version of WOFD introduced here is a sound adaptive numerical method that can reduce the computation work and storage requirements for a given calculation, thereby yielding solutions that can more closely approximate the physically correct solution on any given generation of computers.

Acknowledgments. We thank Dr. Mitsudera for his interest and stimulating discussions during the course of this study. SOEST Contribution No. 4989. IPRC Contribution No. 14.

REFERENCES

Beckmann, A., C. W. Böning, C. Köberle, and J. Willebrand, 1994: Effects of increased horizontal resolution in a simulation of the North Atlantic Ocean. *J. Phys. Oceanogr.,* **24,** 326–344.

Boyd, J. P., 1998: High order models for the nonlinear shallow water wave equations on the equatorial beta-plane with application to Kelvin wave frontogenesis. *Dyn. Atmos. Oceans,* **28,** 69–91.

Daubechies, I., 1988: Orthonormal basis of compactly supported wavelets. *Commun. Pure Appl. Math.,* **41,** 909–996.

Erlebacher, G., M. Y. Hussaini, and L. Jameson, 1996: *Wavelets: Theory and Applications.* Oxford University Press, 510 pp.

Fox, A., and S. J. Maskell, 1995: Two way interactive nesting of primitive equation ocean models with topography. *J. Phys. Oceanogr,* **25,** 2977–2996.

Ginis, I., A. Richardson, and L. M. Rothstein, 1998: Design of a multiply nested primitive equation ocean model. *Mon. Wea. Rev.,* **126,** 1054–1079.

Jameson, L., 1993a: Wavelets and numerical methods. Ph.D. thesis, Brown University, 256 pp. [Available from Division of Applied Mathematics, Brown University, Providence, RI 02912.]

——, 1993b: On the wavelet based differentiation matrix. *J. Sci. Comput.,* **8,** 267–305.

——, 1994: On the wavelet-optimized finite difference method. ICASE Report 94-9, NASA CR-191601, 35 pp. [Available from ICASE, NASA Langley Research Center, M. S. 132C, 3 West Reid St., Building 1152, Hampton, VA 23681-2199; available online at http://www.icase.edu]

——, 1996a: On the differentiation matrix for Daubechies-based wavelets on an interval. *SIAM J. Sci. Comput.,* **17,** 498–516.

——, 1996b: Wavelet-based grid generation. ICASE Rep. 96-59, NASA CR-201609, 33 pp. [Available from ICASE, NASA Langley Research Center, M. S. 132C, 3 West Reid St., Building 1152, Hampton, VA 23681-2199; available online at http://www.icase.edu]

——, 1998: A wavelet-optimized, very high order adaptive grid and order numerical method. *SIAM J. Sci. Comput.,* **19,** 1980–2013.

Kagimoto, T., and T. Yamagata, 1997: Seasonal transport variations of the Kuroshio: An OGCM simulation. *J. Phys. Oceanogr.,* **27,** 403–417.

McClean, J. L., A. J. Semtner, and V. Zlotnicki, 1997: Comparisons of mesoscale variability in the Semtner and Chervin ¼° model, the Los Alamos Parallel Ocean Program ⅙° model, and TOPEX/Poseidon data. *J. Geophys. Res.,* **102,** 25 203–25 226.

Oey, L., and Chen, P., 1992: A Nested-Grid Ocean Model: With application to the simulation of meanders and eddies in the Norwegian Coastal Current. *J. Geophys. Res.,* **97,** 20 063–20 086.

Philander, S. G. H., T. Yamagata, and R. C. Pacanowski, 1984: Unstable air–sea interactions in the Tropics. *J. Atmos. Sci.,* **41,** 604–613.

Sanderson, B. G., 1998; Order and resolution for computational ocean dynamics. *J. Phys. Oceanogr.,* **28,** 1271–1286.

Spall, M. A., and W. R. Holland, 1991: A nested primitive equation model for oceanic application. *J. Phys. Oceanogr.,* **21,** 205–220.

Zhang, D., H. Chang, N. Seeman, T. T. Warner, and J. M. Fritsch, 1986: A two-way interacting nesting procedure with variable terrain resolution. *Mon. Wea. Rev.,* **114,** 1330–1339.