Journal of Computational Physics **181,** 705–728 (2002)
doi:10.1006/jcph.2002.7155

Accuracy and Nonoscillatory Properties of Enslaved Difference Schemes

Don A. Jones,* Len G. Margolin,† and Andrew C. Poje‡

*Department of Mathematics, Arizona State University, Tempe, Arizona 85287-1804; †Center for Nonlinear Studies, Los Alamos National Laboratory, Mail Stop B258, Los Alamos, New Mexico 87545; and ‡Department of Mathematics, College of Staten Island, CUNY, Staten Island, New York
E-mail: dajones@math.la.asu.edu

Received July 6, 2001; revised January 8, 2002

We describe a general method for modifying a given finite-difference scheme by representing the effects of the subgrid scales in terms of the resolved scales through enslaving. The new scheme is more accurate in certain parameter regimes while retaining the time-step stability of the original scheme. We consider two general enslaving relations: an approximate enslaving based on truncation analysis and an *exact* enslaving based on the dynamics of the governing equation. We find that the modified schemes based on the *exact* enslaving eliminate unphysical oscillations, producing monotone solutions even when the original difference schemes do not, have this property. We offer a truncation analysis to justify this property. We apply our enslaving technique to advection–diffusion equations in both one and two spatial dimensions.

1. INTRODUCTION

In previous works [3–7], we have described a procedure, based on enslaving, that improves both the accuracy and computational efficiency of a given finite difference scheme. The increase in accuracy is the result of estimating the effects of the small scales that are unresolved on the mesh in terms of the larger resolved scales. The modified, or *enslaved* scheme, is actually constructed to approximate the solution that the original scheme would generate on a finer mesh. Although the new scheme is more expensive to compute, we have found that in many parameter regimes the added expense is more than compensated by the increase in accuracy.

The idea of enslaving the unresolved scales of motion to the resolved scales has been shown to be effective in several settings. In [4] and [6] we applied the method to the one-dimensional Burgers' Equation with time-dependent forcing. The enslaving relation between the large and small scales was based on truncation analysis and did not depend

0021-9991/02 $35.00

directly on the dynamics of the governing equations. This *approximate enslaving* results in a modified scheme that is never less accurate, has the same temporal stability, and maintains the same memory requirements as the original scheme. In [5] and [7] we showed that the underlying idea of enslavement could be applied to flows in which dissipation is not part of the dominant balance, e.g., such as the circulations of the ocean and the atmosphere. In these papers we constructed several enslaved schemes and applied them to a variety of geophysical flows including the wind-driven double-gyre problem, which is a shallow-fluid flow in an idealized ocean basin. In each case, we demonstrated quantitatively an improvement in computational efficiency.

Our method should not be viewed as a procedure to construct a scheme, but rather as one to improve a given scheme. Our implicit assumption is that the original scheme is appropriate for the given problem. The only claim we make is that our construction leads to a "better" scheme than the one from which we started. Specifically, it is better in the sense that less CPU is required to achieve a desired level of accuracy (i.e., the computational efficiency). Since stability considerations typically dictate the time-step decrease as some power of the grid spacing, the excess cost of enslaving is more than compensated by our ability to use the same time step in the enslaved scheme as in the original scheme while achieving the accuracy of a more finely resolved mesh.

In this paper we examine a new form of the enslaving relation that we construct by ignoring certain time derivatives in the governing PDE. This closure leads to an algebraic relation between the resolved and unresolved scales of motion that we solve exactly. We will compare this *exact enslaving* to the approximate enslaving that results from truncation analysis. While the approximate enslaving is simpler to implement and is generally more computationally efficient, the exact enslaving has the feature that it eliminates unphysical oscillations in the numerical solution. Both enslavings reproduce the accuracy of a standard scheme computed on a mesh twice as fine in parameter regimes where the time derivatives are not part of the principal balance.

In the next section we recall some specifics of the construction of an enslaved difference scheme. In Section 3 we derive both the approximate and exact enslaving for a pair of one-dimensional PDEs. Numerical results are provided in this section along with a comparison of the accuracy and monotonicity properties of the two enslavings. In Section 4 we generalize the same procedure to a two-dimensional advection–diffusion equation and apply the approximate enslaving to the shallow-water equations.

2. BACKGROUND

To construct an enslaved scheme, we assume that we are given a discretized finite-difference equation that approximates the nodal values of the solution of a PDE to some order of accuracy. Our goal is to modify this discretization so as to produce solutions of higher accuracy. Of course, one can always improve the accuracy of a well-posed numerical scheme by increasing the resolution. Thus, an important aspect of our modification is that the new equations should be computationally more efficient; i.e., the modified scheme should require less CPU time than the original scheme to produce a solution of a specified level of accuracy.

Our strategy is to construct a modified scheme on a coarse mesh that reproduces the accuracy of the original scheme on a twice-fine mesh. Modifications based on this strategy,

even if completely successful, cannot improve the order of accuracy. That is, the error of the original scheme on the finer mesh will have the same functional dependence on the grid resolution as the original scheme, but with a smaller coefficient. This feature of the enslaving is explicitly exhibited in Section 3.2.

The increased efficiency results from the relationship between the allowable time step and the given grid spacing in the fully discretized finite-difference scheme. Stability typically dictates that $\Delta t \sim \Delta x^p$. For example, the constant p would be 2 for explicit time-stepping algorithms employing the usual three-point diffusion operator in one spatial dimension. In this case, refining the spatial resolution by a factor of 2 implies an eight-fold increase in the computational work for a one-dimensional problem. We will show that the time step of the enslaved scheme is governed by the spatial resolution of the coarse mesh. Thus, we can improve the accuracy of simulation without reducing the computational time step.

Schematically, our implementation of the enslaving strategy involves the following steps:

1. Define a transformation of variables that maps the degrees of freedom of the fine mesh onto the coarse mesh. Half the variables on the coarse mesh represent the average solution in a cell—the large scales—and the other half represent smaller scales that will be unresolved on the coarse mesh.
2. Transform the original scheme on the twice-fine mesh to the new variables on the coarse mesh and form the evolution equations for the new variables.
3. Apply a closure assumption to the new evolution equation for the small scales. This closure yields a diagnostic relation between the small scales and the large scales.
4. Solve the diagnostic equation to express the small scales in terms of the larger scales. This is the enslaving relation.
5. Implement this relation in the new equation for the large scales. The final result is our improved discretization.

2.1. Coordinate Transformations

To illustrate the first step, we consider a one-dimensional problem where we apply a given difference scheme on a mesh whose cells are $\frac{\Delta x}{2}$ in length (see Fig. 1). On this fine mesh, there are approximately $2N$ cells (depending on the precise relation between the fine and coarse meshes) and therefore $2N$ degrees of freedom. We label these $2N$ degrees of

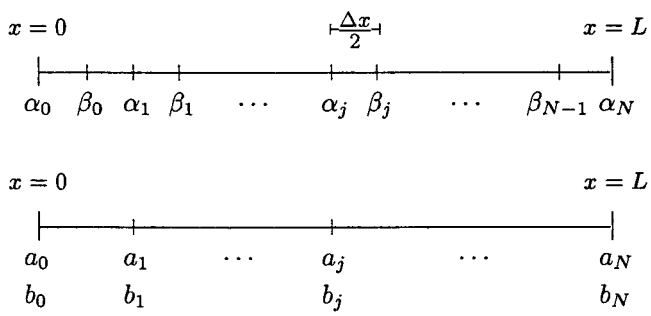

FIG. 1. The fine mesh (top) and the coarse mesh.

freedom alternately with either α or β defined by

$$\alpha_i \approx u(i\,\Delta x)$$
$$\beta_i \approx u\left((2i+1)\frac{\Delta x}{2}\right).$$

(Note that α and β are the same variable evaluated at even and odd nodes; this notational distinction is made only to facilitate our description of the variable transformation.)

The idea now is to define a change of variables that transforms the calculation of $2N$ nodal values (α, β) of the solution on the fine grid to the calculation of N nodal values a_i and N approximations to the spatial derivative b_i on the coarse mesh (Fig. 1). Namely, we want to define a_i and b_i such that

$$a_i \approx u(i\,\Delta x)$$
$$b_i \approx \Delta x \left.\frac{\partial u}{\partial x}\right|_{x=i\,\Delta x}.$$

In order to implement the change of variables, we need to express α and β as functions of a and b. Given the above interpretation of a and b, the first difference of b_i should approximate the second difference of the solution on the fine mesh, $\alpha_{i+1} + \alpha_i - 2\beta_i = b_{i+1} - b_i$. With this motivation, we choose

$$\begin{aligned} \alpha_i &= a_i \\ \beta_i &= \frac{(a_{i+1} + a_i)}{2} + \frac{(b_i - b_{i+1})}{2}, \end{aligned} \tag{2.1}$$

for $0 \le i \le N-1$. The relations in (2.1) provide an explicit definition of a_n and an implicit definition b_n up to a constant (for instance, b_0 or b_N).

As shown in [4], when α_i and β_i approximate sufficiently smooth solutions, the above definition implies

$$b_i = \frac{1}{4}\frac{\partial u(i\,\Delta x)}{\partial x}\Delta x + \mathcal{O}(\Delta x^3). \tag{2.2}$$

Notice that $\frac{b}{a} \sim \mathcal{O}(\Delta x)$. Thus, b_i is representative of the small scales (in both amplitude and spatial extent) while a_i represents the large scales of the solution.

2.2. The Enslaving

Ultimately, given evolution equations for α_i (and equivalently for β_i), the enslaved difference scheme consists of equations for a_i only. Hence, we require diagnostic enslaving relations for the b_i values in terms of the a_i. In this section, we consider two enslavings. The first is the simpler to implement and is based only on truncation error. Equation (2.2) suggests we can approximate b_i by

$$b_i = \frac{a_{i+1} - a_{i-1}}{8}.$$

This enslaving of b has been used successfully in the context of the rotating-shallow-water equations in [5] and [7]. It is the more efficient and the simpler to implement.

An explicit exact enslaving can also be constructed from the underlying PDE. Frequently, the evolution equation for a only involves the quantity $b_{i+1} - b_i$. An expression for this quantity may be obtained by neglecting certain time derivatives. Considering transport equations that include the Laplacian operator explicitly, the general evolution equation at the β_i node has the form

$$\frac{d\beta_i}{dt} - \frac{4\nu}{\Delta x^2}(b_{i+1} - b_i) = R(a_i, b_i),$$

where R represents the remainder of the equation including discretized nonlinear and forcing terms. Neglecting both time derivative and correction terms of $\mathcal{O}(\Delta x^2)$ in $R(a_i, b_i)$ results in the enslaving

$$b_{i+1} - b_i = -\frac{\Delta x^2}{4\nu} R(a_i, 0). \tag{2.3}$$

Enslaving through the dissipative term is only one possibility. It is also possible to enslave through damping terms or rotational terms in a two-dimensional setting. As shown below, the exact enslaving produces a monotone scheme, even when the original scheme does not have this property.

Once the enslaving $b(a)$ is implemented, an evolution equation for a_i is formed. Any time-stepping method may be chosen to advance the a_i. However, the use of multipass time integrators introduces complications in the enslaving process. To construct the enslaving for a two-pass scheme of the form

$$\begin{aligned} \tilde{u}^n_{i,j} &= P_1\big(u^n_{i,j}\big), \\ u^{n+1}_{i,j} &= P_2\big(\tilde{u}^n_{i,j}\big), \end{aligned}$$

we compute $\tilde{a}^n_{i,j}$ by writing $P_1(a^n, b^n)$, where the b^n are known functions of a^n given by the enslaving. The resulting $\tilde{a}_{i,j}$ could then be used to calculate $\tilde{b}$, needed in P_2. In practice, however, we have found that $b^n_{i,j}$ is sufficiently close to $\tilde{b}^n_{i,j}$ to be used in both passes. This leads to a simplified algorithm and modest improvements in efficiency.

2.3. Computational Efficiency

The computational advantage of the new scheme on a coarse grid over the original scheme on the fine grid results from maintaining the time-step restrictions of the coarse grid. For physical systems where diffusion is only one of several processes modeled, the time step of explicit integration methods are limited for numerical stability by

$$\Delta t \le C \Delta x^p,$$

where p is generally 1 or 2 and C is a constant that depends on the solution but not Δx.

Since the enslaved scheme is derived from the given scheme on the twice-fine mesh, we expect that the enslaved scheme will have at most the cost per cycle of the original scheme computed on the twice-fine mesh. In practice, we observe that the enslaved scheme is about 80% of that cost. Thus, if our assumptions concerning the time derivatives are correct and if the enslaved scheme on the coarse mesh has the accuracy of the given scheme on the

fine mesh, there will be a 50% decrease in CPU requirements over the original scheme when the time step is limited by the Courant condition ($p = 1$). In the worst case (i.e., the enslaved scheme is only as accurate as the original coarse-mesh scheme), the calculation of the enslaving relations could result in a twofold increase in computational expense. The computational efficiency of the enslaving procedure is studied in [4].

3. ONE-DIMENSIONAL APPLICATIONS

In this section we illustrate the ideas and the generality of the enslaving procedure outlined in the previous section. The enslavings are applied to several finite difference approximations of one-dimensional PDEs.

3.1. Burgers' Equation

As a first example, we consider Burgers' equation with external forcing.

$$\begin{aligned} \frac{\partial u}{\partial t} - \lambda \frac{\partial^2 u}{\partial x^2} + u \frac{\partial u}{\partial x} &= f(x,t), \\ u(x,0) &= u_0(x), \end{aligned} \tag{3.1}$$

where the viscous coefficient $\lambda > 0$ is given, and $u_0(x)$ is a given smooth function satisfying homogeneous Dirichlet boundary conditions. We assume the second-order accurate spatial discretization

$$\frac{du_i}{dt} - \frac{\lambda}{\Delta x^2}(u_{i+1} - 2u_i + u_{i-1}) + \theta \frac{u_{i+1}^2 - u_{i-1}^2}{4\Delta x} + (1-\theta)\frac{u_i(u_{i+1} - u_{i-1})}{2\Delta x} = f_i, \tag{3.2}$$

where the parameter θ mixes advective and flux form approximations of the transport terms $-0 \le \theta \le 1$. The difference equations on the fine mesh (α, β) can be written as

$$\begin{aligned} \frac{d\alpha_i}{dt} &= \frac{4\lambda}{\Delta x^2}(\beta_i - 2\alpha_i + \beta_{i-1}) - \theta \frac{\beta_i^2 - \beta_{i-1}^2}{2\Delta x} - (1-\theta)\frac{\alpha_i(\beta_i - \beta_{i-1})}{\Delta x} + f_i \\ \frac{d\beta_i}{dt} &= \frac{4\lambda}{\Delta x^2}(\alpha_{i+1} - 2\beta_i + \alpha_i) - \theta \frac{\alpha_{i+1}^2 - \alpha_i^2}{2\Delta x} - (1-\theta)\frac{\beta_i(\alpha_{i+1} - \alpha_i)}{\Delta x} + f_{i+1/2}. \end{aligned} \tag{3.3}$$

At this point the relations given in (2.1) could be used to derive an evolution equation for a_n. However, the appearance of b in the Laplacian term in undesirable, since it would adversely affect the time-step criteria. Indeed, the discrete Laplacian at the α_n node is

$$\frac{4(\beta_i - 2\alpha_i + \beta_{i-1})}{\Delta x^2} = \frac{2(a_{i+1} - 2a_i + a_{i-1})}{\Delta x^2} - \frac{2(b_{i+1} - b_{i-1})}{\Delta x^2}.$$

To avoid the appearance of b in the Laplacian, we will instead approximate a_i as the average

$$a_i \approx \frac{\alpha_i}{2} + \frac{\beta_i + \beta_{i-1}}{4}, \tag{3.4}$$

and then take corresponding linear combinations of the equations in (3.3). In the final scheme, b will only be constructed approximately as a function of a. Therefore, we have a choice in which term to place this error; either in the dissipative term or in the time

derivatives. Since our approximations to b assume small time derivatives, it is preferable to construct an enslaved scheme that leaves the dissipative term independent of b.

We set

$$\langle f\rangle_i = \frac{f_i}{2} + \frac{f_{i-1/2} + f_{i+1/2}}{4}, \quad \langle f'\rangle_i = \frac{f_{i+1/2} - f_{i-1/2}}{4}$$

and

$$A_i = a_i - \left(\frac{\alpha_i}{2} + \frac{\beta_i + \beta_{i-1}}{4}\right) = \frac{1}{8}\left(\delta_x b_i - \delta_x^2 a_i\right),$$

where

$$\delta_x^2 h_i := (h_{i+1} - 2h_i + h_{i-1}), \quad \delta_x h_i := h_{i+1} - h_{i-1}.$$

Now the term A_i term is of order $\alpha_i \Delta x^2$. We have already dropped terms of the order $\frac{d\alpha}{dt}$ in our closure, and so we are also justified in dropping $\frac{dA_i}{dt}$. Then the enslaved equation for a_i is

$$\begin{aligned}\frac{da_i}{dt} &- \frac{\lambda}{\Delta x^2}(a_{i+1} - 2a_i + a_{i-1}) + (2+\theta)\frac{a_{i+1}^2 - a_{i-1}^2}{16\Delta x} + (2-\theta)\frac{a_i(a_{i+1} - a_{i-1})}{8\Delta x} \\ &- \frac{1}{8\Delta x}((a_{i+1} + a_i)(b_{i+1} - b_i) - (a_i + a_{i-1})(b_i - b_{i-1})). \\ &= \langle f\rangle_i. \end{aligned} \tag{3.5}$$

First, we consider the approximate enslaving based purely on truncation analysis

$$b_i = \frac{a_{i+1} - a_{i-1}}{8}. \tag{3.6}$$

As shown in [4] this enslaving produces a scheme that possesses a number of desirable properties. First, when the solution is near steady state, the enslaved scheme has the accuracy of the original scheme (3.2) computed on a twice-fine grid. Second, we observe computationally that the enslaved scheme is never less accurate than the original scheme, regardless of the temporal behavior of the solution. Finally, the time-step restrictions for any temporal discretization of (3.2) and (3.5) are the same.

Next we construct the exact enslaving for Burgers' equation. To implement the enslaving we need to find $b_{i+1} - b_i$ in (3.5) as a function of a_i. We find this quantity by applying the change of variables given in (2.1) to the differencing of Burgers' equation at the β_i node. We derive

$$\frac{d\beta_i}{dt} - \frac{4\lambda}{\Delta x^2}(b_{i+1} - b_i) + \frac{a_{i+1}^2 - a_i^2}{2\Delta x} + \frac{1-\theta}{4\Delta x}(a_{i+1} - a_i)(b_i - b_{i+1}) = f_{i+1/2}.$$

Neglecting $d\beta_i/dt$ and keeping only the terms that lead to second-order corrections, we construct the exact enslaving

$$b_{i+1} - b_i = \frac{\Delta x^2}{4\lambda}\left[\frac{a_{i+1}^2 - a_i^2}{2\Delta x} - f_{i+1/2}\right]. \tag{3.7}$$

3.2. A Comparison of Enslavings

In general, the exact enslaving (3.7) is not an efficient enslaving in terms of computational effort, particularly in higher space dimensions. As previously mentioned, the approximate enslaving (3.6) is more efficient, although it uses no information about the dynamics of the b equation. However, a standard truncation analysis reveals that the two enslavings (3.6) and (3.7) are close when time derivatives are small. From the approximate enslaving, (3.6), we find

$$b_{i+1} + b_{i-1} - 2b_i = \frac{1}{4} u_{xxx} \Delta x^3 + \mathcal{O}(\Delta x_5).$$

Using Burgers' Equation, we can write this truncation error as

$$b_{i+1} + b_{i-1} - 2b_i = \frac{\Delta x^3}{8\lambda} (u^2)_{xx} - \frac{\Delta x^3}{4\lambda} \frac{\partial f}{\partial x} + \frac{\Delta x^3}{4\lambda} \frac{\partial u_x}{\partial t} + \mathcal{O}(\Delta x^5).$$

The exact enslaving given by (3.7) approximates

$$b_{i+1} + b_{i-1} - 2b_i = \frac{\Delta x^3}{8\lambda} (u^2)_{xx} - \frac{\Delta x^3}{4\lambda} \frac{\partial f}{\partial x} + \mathcal{O}(\Delta x^5).$$

The two enslavings formally differ by the term $\frac{\Delta x^3}{4\lambda} \frac{\partial u_x}{\partial t}$.

The reason that the enslaved scheme is more accurate than the given scheme in certain regimes becomes clear upon comparing truncation analysis of the two schemes. The spatial truncation error for the standard scheme is

$$T_i^s = \left(\frac{\lambda}{12} u_{xxxx} - \frac{u u_{xxx}}{6} - \theta \frac{u_x u_{xx}}{2} \right) \Delta x^2 + \mathcal{O}(\Delta x^4),$$

and the spatial truncation error for the enslaved scheme using (3.6) is

$$T_i^e = \left(\frac{\lambda}{12} u_{xxxx} - \frac{5 u u_{xxx}}{48} - (3 + 2\theta) \frac{u_x u_{xx}}{16} + \frac{f_{xx}}{16} \right) \Delta x^2 + \mathcal{O}(\Delta x^4). \tag{3.8}$$

If we take two spatial derivatives of Burgers' equation at steady state, we obtain

$$-\lambda u_{xxxx} + 3 u_x u_{xx} + u u_{xxx} = f_{xx}.$$

By replacing f_{xx} in (3.8), we find that the error introduced by averaging the forcing over three grid points interacts with the error of the coarse scheme so that $T_i^e = (1/4) T_i^s$ when time derivatives are small. A similar analysis holds for the exact enslaving (3.7).

One could, of course, reduce the truncation error of the standard scheme by any amount by subtracting appropriate approximations of the truncation error. Such a procedure would, however, adversely affect the time-step restrictions for the stability of the scheme (i.e., decrease the allowable time step). Even if the PDE at steady state is used to change the functional dependence of the truncation error, the resulting modified scheme will likely be less efficient than the original difference scheme. The combination of the change of variables from fine to coarse mesh and the use of the governing PDE at steady enables the enslaving procedure to reduce the truncation error without adversely affecting the original properties of the given difference scheme. In addition, the specific form of the exact enslaving, (3.7),

forces an interaction between the truncation error of the nonlinear term and the dissipative term in a way that results in a monotone scheme, even when the original scheme is not monotone. This feature will be proved in the next section.

An an illustration of the properties of the two enslavings, we force Burgers' Equation so that $u(x,t) = \eta \sin(\pi x) + \varepsilon \sin(\pi x)(x\cos(\omega t) + (1-x)\sin(\omega t))$ is an exact solution (cf. [4]). The parameters (η, ϵ, ω) allow us to vary the importance of the temporal derivatives. We choose $\eta = \varepsilon = 0.5$ and $\omega = 2\pi$, which produces $\mathcal{O}(\pi)$ contributions from the time derivative. The standard scheme is an improved Euler method time discretization of (3.2) with $\theta = 2/3$, $\lambda = 0.05$, $\Delta t = 0.001$ and $\Delta x = 1/60$.

As a diagnostic, we compare the error of the standard scheme to that of the enslaved scheme. We define the ratio of the L^∞ norm of the difference of the exact solution, $U(t)$, and the standard scheme, $u(t)$, to the difference of the exact solution and the enslaved scheme, $v(t)$:

$$R(t) = \frac{\|u(t) - U(t)\|_\infty}{\|v(t) - U(t)\|_\infty}.$$

For a given second-order accurate spatial differencing, the error ratio will approach $R(t) = 4$ when the enslaved scheme nearly approximates the the solution on the fine grid.

Figure 2 indicates the error ratio as a function of time for two different enslavings: the approximate enslaving given in (3.6) and the exact enslaving given by (3.7). The average value of the ratio, $R(t)$, over the run is 2.86 for the approximate enslaved scheme and 2.04 for the exact enslaved scheme.

For either choice of enslavings, the accuracy of the modified scheme is markedly increased over that of the original scheme for most of the time period. The instantaneous improvement is often well above the theoretical value of $R(t) = 4$ while the time-averaged ratio is close to this value. Notably, $R(t) > 1$ throughout indicating that the enslaved scheme solutions are never less accurate than the original, coarse grid scheme solutions, even during periods of fast oscillation.

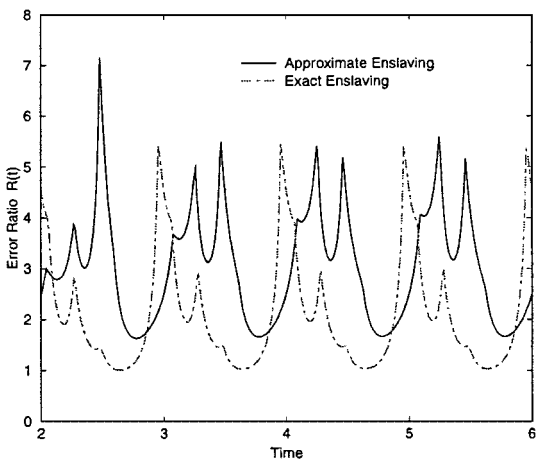

FIG. 2. Burgers' Equation with a time-dependent force and known exact solution. The error ratio is plotted using the approximate enslaving, (3.6), and the exact enslaving, (3.7).

3.3. Nonoscillatory Properties

In addition to improving the accuracy of the solution, the exact enslaving relations can further improve the stability (i.e., relax the time step restrictions) and reduce the unphysical oscillations of a given scheme. To investigate the damping of spurious numerical oscillations, we first consider the exact enslaving of the linear advection-diffusion equation:

$$\frac{\partial u}{\partial t} + C\frac{\partial u}{\partial x} = \nu\frac{\partial^2 u}{\partial x^2}, \tag{3.9}$$

where C is a constant positive velocity and ν is a small but positive diffusion coefficient. A standard, second-order accurate spatial differencing of (3.9) on the (α, β) grid is

$$\begin{aligned}\frac{d\alpha_i}{dt} &= \frac{4\nu}{\Delta x^2}(\beta_i - 2\alpha_i + \beta_{i-1}) - \frac{C}{\Delta x}(\beta_i - \beta_{i-1})\\ \frac{d\beta_i}{dt} &= \frac{4\nu}{\Delta x^2}(\alpha_{i+1} - 2\beta_i + \alpha_i) - \frac{C}{\Delta x}(\alpha_{i+1} - \alpha_i).\end{aligned} \tag{3.10}$$

Following the program defined above, we derive the following evolution equation for a:

$$\frac{da_i}{dt} = \frac{\nu}{\Delta x^2}(a_{i+1} - 2a_i + a_{i-1}) - \frac{C}{2\Delta x}(a_{i+1} - a_{i-1}) + \frac{C}{4\Delta x}\left(\delta_x^2 b_i\right). \tag{3.11}$$

To implement the enslaving in (3.11), we need to replace $\delta_x^2 b_i$ by a function of a_i. We find this relationship by applying the change of variables given in (2.1) to the equation for β_i in (3.10); specifically, by neglecting $d\beta_i/dt$, we construct the enslaving

$$b_{i+1} - b_i = \frac{C\Delta x}{4\nu}(a_{i+1} - a_i). \tag{3.12}$$

Inserting (3.12) into the equation for a_i, we derive the enslaved scheme

$$\frac{da_i}{dt} = \frac{\nu_e^+}{\Delta x^2}(a_{i+1} - 2a_i + a_{i-1}) - \frac{C}{2\Delta x}(a_{i+1} - a_{i-1}), \tag{3.13}$$

where the augmented diffusion coefficient

$$\nu_e^+ = \nu\left(1 + \frac{C^2\Delta x^2}{16\nu^2}\right). \tag{3.14}$$

For explicit time-stepping schemes, one can show that solutions of (3.10) solved on a mesh of size Δx are monotonic (nonoscillatory) only when the condition $\mathrm{Re}_{\Delta x} = \frac{C\Delta x}{\nu} \le 2$ is satisfied, (cf. [1], [10]). Using the above definition of the numerical diffusion, the grid Reynolds number for the enslaved scheme is

$$\mathrm{Re}_e = \frac{C\Delta x}{\nu_e^+} = \mathrm{Re}_{\Delta x}\left(\frac{1}{1 + \left(\frac{\mathrm{Re}_{\Delta x}}{4}\right)^2}\right).$$

In Fig. 3 (left panel) we plot the enslaved grid Reynolds number versus the grid Reynolds number on the coarse mesh. A grid Reynolds less than two implies the scheme is monotone. The enslaved scheme is evidently monotone for all values of $\mathrm{Re}_{\Delta x}$. The right panel in

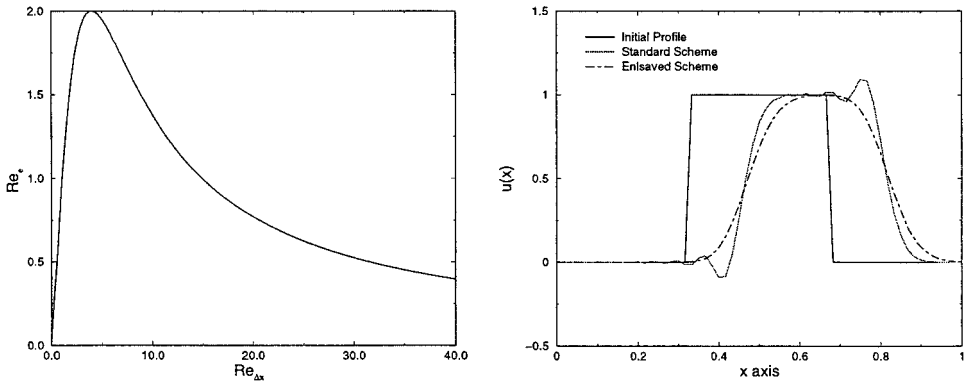

FIG. 3. The relation between the grid Reynolds number in the standard and enslaved schemes (left). The solutions (right) of the advection–diffusion equation with constant advective velocity use forward Euler time integration and a grid Reynolds number of $\mathrm{Re}_{\Delta x} = 8.0$ corresponding to a enslaved grid Reynolds number of 1.6.

Fig. 3 shows the solutions of the standard and enslaved schemes when $\mathrm{Re}_{\Delta x} = 8.0$. Periodic boundary conditions are imposed, $\nu = 0.0002$, $C = 0.1$, and $\Delta x = 1/60$, and an improved Euler method is used for the time integration with $\Delta t = 0.01$. Note that the standard scheme has unphysical oscillations while the enslaving relation introduces sufficient numerical diffusion to ensure monotonicity of the solution.

To illustrate the effects of the enslaving procedure on a scheme that is initially monotone, we next consider the upwind differencing of (3.9)

$$\begin{aligned} u_i^{n+1} &= u_i^n + \Delta t \frac{\nu}{\Delta x^2}\left(u_{i+1}^n - 2u_i^n + u_{i-1}^n\right) - \Delta t \frac{C}{\Delta x}\left(u_i^n - u_{i-1}^n\right) \\ &= u_i^n\left[1 - \frac{2\nu\Delta t}{\Delta x^2} - \frac{C\Delta t}{\Delta x}\right] + u_{i+1}^n\left[\frac{\nu\Delta t}{\Delta x^2}\right] + u_{i-1}^n\left[\frac{\nu\Delta t}{\Delta x^2} + \frac{C\Delta t}{\Delta x}\right]. \end{aligned} \tag{3.15}$$

We have now assumed $C > 0$. Because the equations are linear, the scheme is monotone if and only if each of the coefficients by the u_i^n in (3.15) is positive [1]. The last two coefficients are clearly positive. Positive definiteness of the first coefficient requires

$$2 + \mathrm{Re}_{\Delta x} < \frac{\Delta x^2}{\nu \Delta t}. \tag{3.16}$$

Next we apply the enslaving procedure. In this setting (2.3) becomes

$$b_{i+1} - b_i = \frac{\mathrm{Re}_{\Delta x}}{4}(a_{i+1} - a_i).$$

The semi-discrete enslaved scheme is

$$\frac{da}{dt} = \frac{\nu_e^-}{\Delta x^2}(a_{i+1} - 2a_i + a_{i-1}) - \frac{C}{\Delta x}(a_i - a_{i-1}),$$

where

$$\nu_e^- = \nu\left(1 + \left(\frac{\mathrm{Re}_{\Delta x}}{4}\right)^2 - \left(\frac{\mathrm{Re}_{\Delta x}}{4}\right)\right).$$

For $\mathrm{Re}_{\Delta x} \in (0, 4)$, the enslaving relation actually reduces the effective numerical viscosity

present in the upwinding scheme. Upon discretizing the time, we find

$$a_i^{n+1} = a_i^n \left[1 - \frac{2\nu_e^- \Delta t}{\Delta x^2} - \frac{C\Delta t}{\Delta x}\right] + a_{i+1}^n \left[\frac{\nu_e^- \Delta t}{\Delta x^2}\right] + a_{i-1}^n \left[\frac{\nu_e^- \Delta t}{\Delta x^2} + \frac{C\Delta t}{\Delta x}\right].$$

Again, the scheme will be monotone provided all of the coefficients are positive. Positive definiteness of the first coefficient requires

$$2\frac{\nu_e^-}{\nu} + \mathrm{Re}_{\Delta x} < \frac{\Delta x^2}{\nu \Delta t}. \tag{3.17}$$

The second and third coefficients are clearly positive.

Figure 4 shows a plot of the maximum allowable time step as a function of $\mathrm{Re}_{\Delta x}$ for fixed Δx and ν (the velocity is variable, and hence $\mathrm{Re}_{\Delta x}$ varies as well). The maximum allowable time step is obtained by solving for Δt as a function of $\mathrm{Re}_{\Delta x}$ in the inequalities (3.16) and (3.17). We see that, for $0 < \mathrm{Re}_{\Delta x} < 4$, the requirement on Δt for the inequalities to hold is less severe for the enslaved scheme. Thus, the enslaving procedure applied to an upwind differencing has led to a monotone scheme with less severe time-step restrictions. The right panel in Fig. 4 illustrates a case where the upwind scheme is unstable while the enslaved scheme is not. The initial data is a top-hat function with height unity, centered at the middle of the domain. Homogeneous Dirichlet boundary conditions are imposed. We choose $\Delta x = 1/60$, $\nu = 0.01$, $C = 0.495$, and $\Delta t = 0.01$. Both schemes are integrated to $T = 1.0$.

The case where $\mathrm{Re}_{\Delta x} > 4$ merits further discussion. The apparent increase in the effective viscosity is not a failure of the enslaving concept, but rather results from using the closure (2.3), which is not accurate in this parameter range. In particular, the problem arises from neglecting the b-dependence in the remainder function. When the full remainder function $R(a, b)$ is used in (2.3) and the resulting implicit relation is solved for $(b_{i+1} - b_i)$, then the effective viscosity is decreased *for all* values of $\mathrm{Re}_{\Delta x}$. This generalization is easily implemented in one dimension, but solving the implicit relation in two or more spatial dimensions becomes prohibitively expensive. It is unlikely that one would seriously consider using a donor cell as a basic scheme since the enslaved scheme would remain only first-order accurate. However, the enslaving based on truncation error would remain an effective alternative.

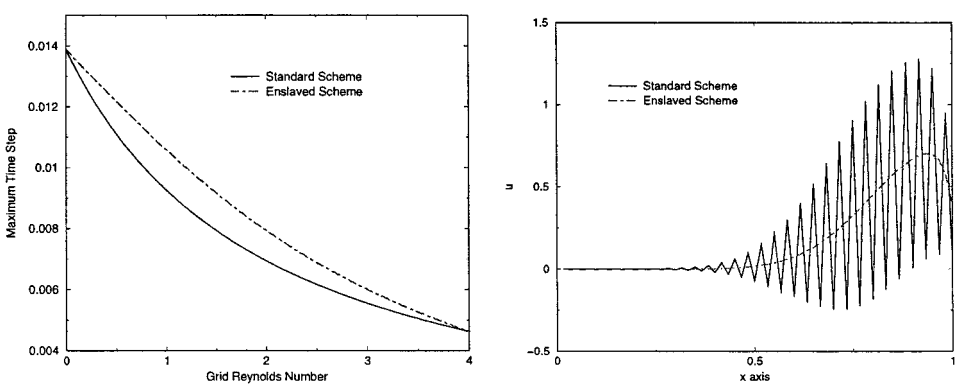

FIG. 4. The maximum allowable time step for the standard and enslaved schemes for fixed grid resolution and diffusion constant (left). Integration of the enslaved and standard schemes for the upwind discretization (right).

The nonoscillatory properties of the exact enslaving are also present in nonlinear PDEs. We demonstrate this analytically and also computationally by considering two examples of Burgers' equation without forcing. To derive the enslaved equations, we use (3.7) to write the correction terms, i.e., the terms in (3.5) involving the slaved quantity b as

$$\frac{-1}{64\lambda}\left[(a_{i+1}+a_n)\left(a_{i+1}^2-a_i^2\right)-(a_i+a_{i-1})\left(a_i^2-a_{i-1}^2\right)\right].$$

We define the operators

$$\delta_x^+ a_i = \frac{a_{i+1}-a_i}{\Delta x}, \quad \delta_x^- a_i = \frac{a_i - a_{i-1}}{\Delta x}.$$

When we use these operators, the expression for the correction term becomes

$$-\frac{\Delta x^2}{64\lambda}\delta_x^+[(a_i+a_{i-1})^2\delta_x^- a_i]. \tag{3.18}$$

The dissipative term in (3.5), $-\lambda\delta_x^+\delta_x^- a_i$, combines with (3.18) to form the effective dissipative term

$$-\delta_x^+\left[\lambda\left(1+\frac{\Delta x^2}{16\lambda^2}\left(\frac{a_i+a_{i-1}}{2}\right)^2\right)\delta_x^- a_i\right].$$

In analogy to (3.14) in the linear case, we define

$$\lambda_i^e := \lambda\left(1+\frac{\Delta x^2}{16\lambda^2}\left(\frac{a_i+a_{i-1}}{2}\right)^2\right).$$

The exact enslaved scheme for Burgers' equation without forcing is given by

$$\frac{da_i}{dt}-\delta_x^+\left(\lambda_i^e\delta_x^- a_i\right)+(2+\theta)\frac{a_{i+1}^2-a_{i-1}^2}{16\Delta x}+(2-\theta)\frac{a_i(a_{i+1}-a_{i-1})}{8\Delta x}=0. \tag{3.19}$$

Analogous to the linear case (see Eq. (3.13)), the enslaving procedure produces a nonlinear numerical viscosity that ensures the nonoscillatory behavior of the solutions. To prove that the enslaved scheme is sign preserving, we assume $a_i^n \geq 0$ for all i and consider a forward Euler discretization of (3.19). The discretization may be written

$$\begin{aligned} a_i^{n+1} = {} & a_i^n\left[1-\frac{\Delta t}{\Delta x^2}\left(\lambda_{i+1}^e+\lambda_i^e\right)\right]+a_{i+1}^n\Delta t\left[\frac{\lambda_{i+1}^e}{\Delta x^2}-\frac{2+\theta}{16\Delta x}a_{i+1}^n-\frac{2-\theta}{8\Delta x}a_i^n\right] \\ & +a_{i-1}^n\Delta t\left[\frac{\lambda_i^e}{\Delta x^2}+\frac{2+\theta}{16\Delta x}a_{i-1}^n+\frac{2-\theta}{8\Delta x}a_i^n\right]. \end{aligned}$$

The first term is required to be nonnegative to ensure the overall stability of the scheme; the third term is clearly nonnegative. We define the local grid Reynolds number

$$\mathrm{Re}_i = \frac{a_i^n\Delta x}{\lambda}.$$

The second term is nonnegative provided

$$1+\frac{1}{16}\left(\frac{\mathrm{Re}_{i+1}+\mathrm{Re}_i}{2}\right)^2-\frac{2+\theta}{16}\mathrm{Re}_{i+1}-\frac{2-\theta}{8}\mathrm{Re}_i \geq 0.$$

It is straightforward, but tedious to demonstrate that the function $f(x, y) = 1 + \frac{(x+y)^2}{64} - (2+\theta)\frac{x}{8} - (2-\theta)\frac{y}{8}$ is nonnegative for all $x, y \geq 0$ and $0 \leq \theta \leq 1$. Thus, the enslaved scheme for Burgers' Equation without forcing is sign preserving.

We further demonstrate the nonoscillatory properties in the following numerical examples. First, we study the kinetic energy spectrum resulting from the relaxation of random initial conditions in the unforced Burgers' Equation. In this experiment we consider only periodic boundary conditions. The initial data is constructed by varying the phases of modes four through eight randomly with uniform distribution. The amplitude is chosen from a Gaussian distribution with the peak at mode six. Thus, the initial data may be written

$$u_0(x) = \sum_{k=-N}^{N} c_k \alpha_1 e^{-\alpha 2(|k|-4)^2} e^{i\theta_k} e^{2\pi i k x},$$

where $\theta_k = \theta_{-k}$ is randomly chosen between $[0, 2\pi]$ and the coefficients $|c_k| = 1$, $4 \leq |k| \leq 8$, and zero otherwise, are chosen so that initial data is real. Each realization decays from these initial conditions to a fixed time when the kinetic energy is stored. By varying the random seed, statistics from 200 independent simulations are obtained. At a fixed resolution, we ensure that both the standard and the exact enslaved schemes have exactly the same sets of initial data.

Figure 5 shows the results of the simulations averaged at time $T = 0.5$, with the parameters $\lambda = 0.00012$, $\alpha_1 = 0.05$, $\alpha_2 = 0.1$, and $\theta = 2/3$. The standard scheme employs an

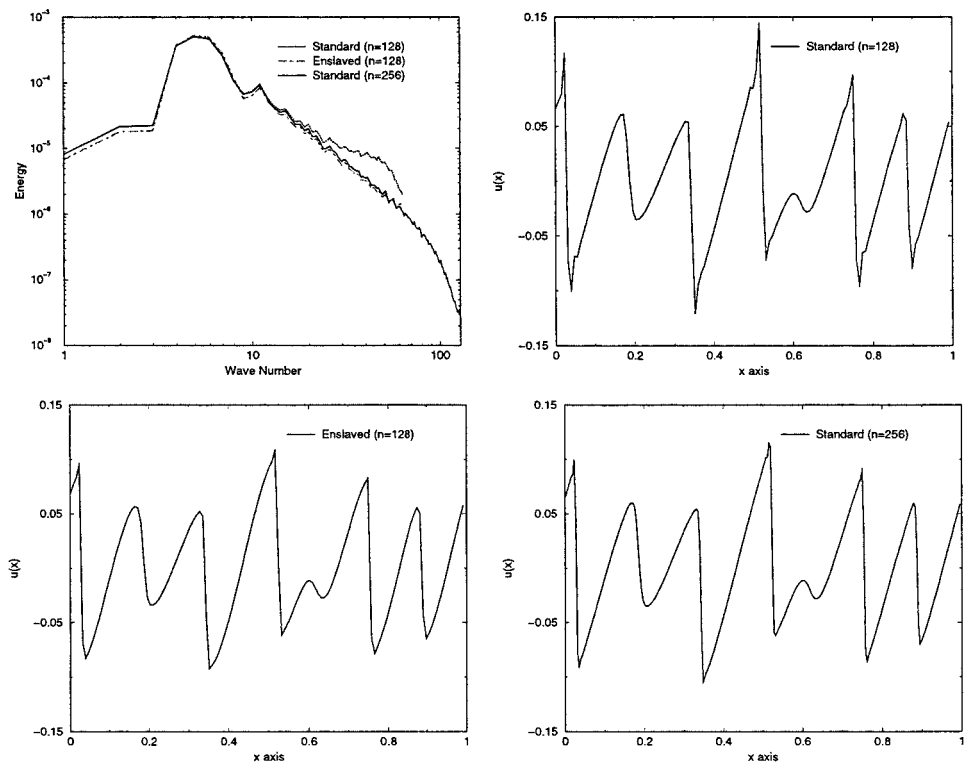

FIG. 5. The upper-left corner shows a comparison of the averaged kinetic energy spectrum with random initial data. The other plots graph the solution of the end of the run.

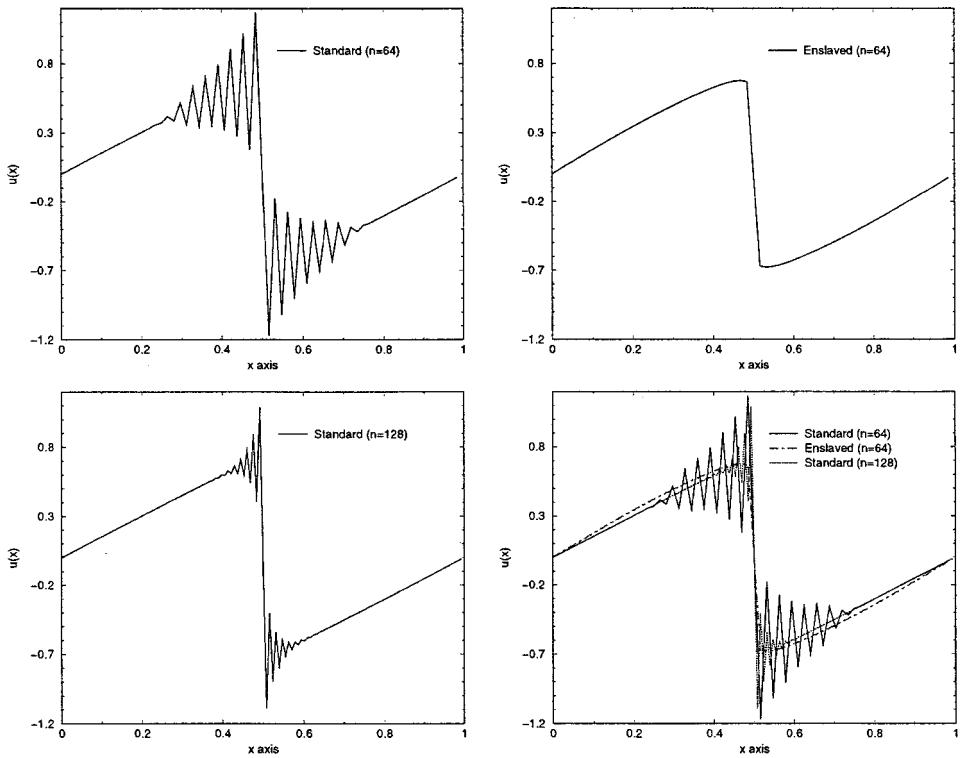

FIG. 6. Burgers' Equation with initial data $u_0 = \sin(2\pi x)$. The standard scheme with $\Delta x = 1/64$ is in the upper-left corner. The upper-right corner shows the enslaved scheme; the lower-left corner shows the twice fine scheme, and the lower right shows all three.

improved Euler method for the time discretization of (3.2) with $\Delta t = 0.001$. The plot in the upper-left corner compares the spectrum of the enslaved scheme to that of the standard scheme at two different resolutions. The remaining plots show the individual solutions at the end of the same realization. Notice the standard scheme on the coarse mesh ($\Delta x = 1/128$) has overshoots while the enslaved scheme does not. The overshoots cause the spurious behavior in the energy spectrum seen in Fig. 5.

For our second example, we again consider Burgers' Equation and assume no external forcing is present. Here we choose the initial data $u_0(x) = \sin(2\pi x)$, and specify the boundary conditions as either periodic or homogeneous Dirichlet (since the initial data is an odd function about the origin, these are equivalent). The standard and enslaved schemes are the same as those used in the previous example. The diffusion constant is $\lambda = 0.0005$, and the resolution of the coarse mesh is $\Delta x = 1/64$. The initial data is allowed to decay; the upper-left plot in Fig. 6 shows the solution at $T = 0.5$. The plot in the upper-right corner is obtained from the enslaved scheme at the same time and resolution. The solution again exhibits the nonoscillatory properties of the enslaved scheme.

4. TWO-DIMENSIONAL APPLICATIONS

In this section we generalize the enslaving procedure to multiple spatial dimensions. The first step in the analysis requires the choice of a change of variables from a twice-fine

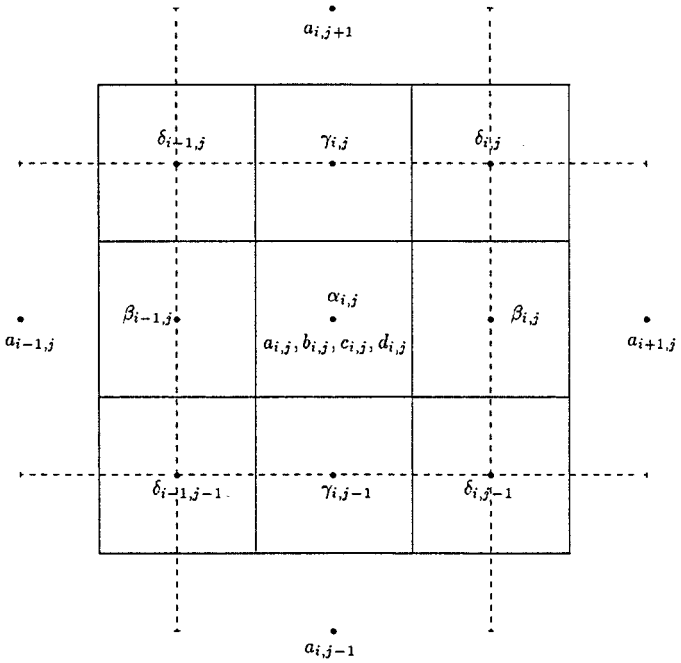

FIG. 7. Fine and coarse meshes for the velocity fields.

mesh to a coarse mesh. We label the fine mesh $\alpha_{i,j}$, $\beta_{i,j}$, $\gamma_{i,j}$ and $\delta_{i,j}$ (see Fig. 7). The new variables, $\alpha_{i,j}$, $b_{i,j}$, $c_{i,j}$, and $d_{i,j}$ are chosen to approximate

$$
\begin{aligned}
a_{i,j} &\approx u(i\Delta x,\, j\Delta y),\\
b_{i,j} &\approx \Delta x \frac{\partial u}{\partial x}\bigg|_{x=i\Delta x, y=j\Delta y},\\
c_{i,j} &\approx \Delta y \frac{\partial u}{\partial y}\bigg|_{x=i\Delta x, y=j\Delta y},\\
d_{i,j} &\approx \Delta y \Delta x \frac{\partial^2 u}{\partial x \partial y}\bigg|_{x=i\Delta x, y=j\Delta y}.
\end{aligned}
$$

Generalizing the one-dimensional procedure, we define the variables a, b, c, and d through the relations

$$
\begin{aligned}
\alpha_{i,j} &= a_{i,j},\\
\beta_{i,j} &= \frac{(a_{i+1,j} + a_{i,j})}{2} + \frac{(b_{i,j} - b_{i+1,j})}{2},\\
\gamma_{i,j} &= \frac{(a_{i,j+1} + a_{i,j})}{2} + \frac{(c_{i,j} - c_{i,j+1})}{2},\\
\delta_{i,j} &= \frac{(a_{i,j+1} + a_{i,j} + a_{i+1,j} + a_{i+1,j+1})}{4} + \frac{(b_{i,j} + b_{i,j+1} - b_{i+1,j} - b_{i+1,j+1})}{4}\\
&\quad + \frac{(c_{i,j} + c_{i+1,j} - c_{i,j+1} - c_{i+1,j+1})}{4} + \frac{(d_{i,j} + d_{i+1,j+1} - d_{i,j+1} - d_{i+1,j})}{4}.
\end{aligned}
\tag{4.1}
$$

The rather complicated expression for $\delta_{i,j}$ is designed to preserve certain properties of the discrete Laplacian. Equations (4.1) could be used directly to construct an evolution equation for α_i. However, this procedure would place $b_{i,j}$ and $c_{i,j}$ in the Laplacian and adversely affect the allowable time step. To avoid this, linear combinations of the equations, similar to (3.4), are formed. By summing areas of overlap in Fig. 7, the linear combinations are found to be

$$\begin{aligned}
a_{i,j} &\approx \frac{a_{i,j}}{4} + \frac{\beta_{i,j} + \beta_{i-1,j}}{8} + \frac{\gamma_{i,j} + \gamma_{i,j-1}}{8}, \quad + \frac{\delta_{i,j} + \delta_{i-1,j} + \delta_{i,j-1} + \delta_{i-1,j-1}}{16}, \\
b_{i,j} &\approx \frac{\beta_{i,j} - \beta_{i-1,j}}{8} + \frac{\delta_{i,j} - \delta_{i-1,j} + \delta_{i,j-1} - \delta_{i-1,j-1}}{16}, \\
c_{i,j} &\approx \frac{\gamma_{i,j} - \gamma_{i,j-1}}{8} + \frac{\delta_{i,j} + \delta_{i-1,j} - \delta_{i,j-1} - \delta_{i-1,j-1}}{16}, \\
d_{i,j} &\approx \frac{(\delta_{i,j} + \delta_{i-1,j-1})}{16} - \frac{(\delta_{i-1,j} + \delta_{i,j-1})}{16}.
\end{aligned} \tag{4.2}$$

4.1. Advection–Diffusion Equation in 2D

To illustrate the performance of the enslaved method in multiple dimensions, we first apply the technique of the previous section to a linear, two-dimensional, advection–diffusion equation. We assume that the boundary conditions are doubly periodic, $\mathbf{C} = (C_1, C_2)$, and ν is a small but positive diffusion coefficient. We consider the PDE

$$\frac{\partial u}{\partial t} + \mathbf{C} \cdot \nabla u = \nu \nabla^2 u. \tag{4.3}$$

We begin by approximating (4.3) on a fine grid containing $4(N \times N)$ nodes (see Fig. 7). We define

$$\begin{aligned}
\delta^2_\alpha &= \frac{4\nu}{\Delta x^2}\{\beta_{i,j} + \beta_{i-1,j} - 4\alpha_{i,j} + \gamma_{i,j} + \gamma_{i,j-1}\}, \\
\delta^2_\beta &= \frac{4\nu}{\Delta x^2}\{\alpha_{i+1,j} + \alpha_{i,j} - 4\beta_{i,j} + \delta_{i,j} + \delta_{i,j-1}\}, \\
\delta^2_\gamma &= \frac{4\nu}{\Delta x^2}\{\delta_{i,j} + \delta_{i-1,j} - 4\gamma_{i,j} + \alpha_{i,j+1} + \alpha_{i,j}\}, \\
\delta^2_\delta &= \frac{4\nu}{\Delta x^2}\{\gamma_{i+1,j} + \gamma_{i,j} - 4\delta_{i,j} + \beta_{i,j+1} + \beta_{i,j}\}.
\end{aligned}$$

A standard differencing of (3.9) on the α, β, γ, and δ grid is given by

$$\begin{aligned}
\frac{d\alpha_{i,j}}{dt} + C_1\frac{\beta_{i,j} - \beta_{i-1,j}}{\Delta x} + C_2\frac{\gamma_{i,j} - \gamma_{i,j-1}}{\Delta x} &= \delta^2_\alpha, \\
\frac{d\beta_{i,j}}{dt} + C_1\frac{\alpha_{i+1,j} - \alpha_{i,j}}{\Delta x} + C_2\frac{\delta_{i,j} - \delta_{i,j-1}}{\Delta x} &= \delta^2_\beta, \\
\frac{d\gamma_{i,j}}{dt} + C_1\frac{\delta_{i,j} - \delta_{i-1,j}}{\Delta x} + C_2\frac{\alpha_{i,j+1} - \alpha_{i,j}}{\Delta x} &= \delta^2_\gamma, \\
\frac{d\delta_{i,j}}{dt} + C_1\frac{\gamma_{i+1,j} - \gamma_{i,j}}{\Delta x} + C_2\frac{\beta_{i,j+1} - \beta_{i,j}}{\Delta x} &= \delta^2_\delta.
\end{aligned} \tag{4.4}$$

Applying the change of variables on a uniform grid and taking the linear combinations defined above, the evolution equation for $a_{i,j}$ becomes

$$\begin{aligned}\frac{da_{i,j}}{dt} &= \frac{\nu}{8\Delta x^2}\{4\otimes + 4\oplus - \delta_x\delta_y^2 b_{i,j} - \delta_y\delta_x^2 c_{i,j}\} - \frac{C_1}{16\Delta x}(\delta_x a_{i,j-1} + 6\delta_x a_{i,j} + \delta_x a_{i,j+1}) \\ &+ \frac{C_1}{32\Delta x}\left(\delta_x^2 b_{i,j-1} + 6\delta_x^2 b_{i,j} + \delta_x^2 b_{i,j+1} + 2\delta_{xy}^2 c_{i,j}\right) \\ &- \frac{C_2}{16\Delta x}(\delta_y a_{i-1,j} + 6\delta_y a_{i,j} + \delta_y a_{i+1,j}) \\ &+ \frac{C_2}{32\Delta x}\left(\delta_y^2 c_{i-1,j} + 6\delta_y^2 c_{i,j} + \delta_y^2 c_{i+1,j} + 2\delta_{xy}^2 b_{i,j}\right).\end{aligned}$$

The new scheme employs a nine-point stencil, using a combination of the two Laplacian operators

$$\begin{aligned}\oplus &= \{a_{i+1,j} + a_{i-1,j} - 4a_{i,j} + a_{i,j+1} + a_{i,j-1}\}, \\ \otimes &= \frac{1}{2}\{a_{i+1,j+1} + a_{i+1,j-1} - 4a_{i,j} + a_{i-1,j+1} + a_{i-1,j-1}\}.\end{aligned}$$

To construct the enslaving we notice the a equation only requires the enslavement of $(b_{i+1,j} - b_{i,j})$ and $(c_{i,j+1} - c_{i,j})$ in terms of the variable a. The enslavings are found in a similar manner as in the one-dimensional framework. We define

$$\delta_x^+ a_{i,j} = a_{i+1,j} - a_{i,j}, \quad \delta_y^+ a_{i,j} = a_{i,j+1} - a_{i,j},$$

and

$$\chi_{i,j} = \delta_x^+(b_{i,j+1} + 2b_{i,j} + b_{i,j-1}) + \delta_y(c_{i,j} + c_{i+1,j}).$$

Applying the change of variables to the $\beta_{i,j}$ equation (4.4) and neglecting higher-order corrections, we find

$$\begin{aligned}&\frac{d\beta_{i,j}}{dt} - \frac{\nu}{\Delta x^2}\left(\left(\delta_y^2 a_{i,j} + \delta_y^2 a_{i+1,j}\right) + 8(b_{i+1,j} - b_{i,j}) - \chi_{i,j}\right) \\ &\quad + \frac{C_1}{\Delta x}\delta_x^+ a_{i,j} + \frac{C_2}{4\Delta x}(\delta_y a_{i,j} + \delta_y a_{i+1,j}) = 0. \qquad (4.5)\end{aligned}$$

Next, we add the differencing of the governing equations at the points $\delta_{i,j}$ and $\delta_{i,j-1}$. Furthermore, we neglect all time derivatives. We derive

$$\chi_{i,j} = \frac{C_1\Delta x}{4\nu}(\delta_x^+ a_{i,j+1} + 2\delta_x^+ a_{i,j} + \delta_x^+ a_{i,j-1}) + \frac{C_2\Delta x}{4\nu}(\delta_y a_{i+1,j} + \delta_y a_{i,j}).$$

Inserting this into Equation (4.5) and again neglecting the time derivatives, we obtain the enslaving

$$\begin{aligned}b_{i+1,j} - b_{i,j} &= -\frac{1}{8}\left(\delta_y^2 a_{i,j} + \delta_y^2 a_{i+1,j}\right) + \frac{C_1\Delta x}{32\nu}(\delta_x^+ a_{i,j+1} + 6\delta_x^+ a_{i,j} + \delta_x^+ a_{i,j-1}) \\ &+ \frac{C_2\Delta x}{16\nu}(\delta_y a_{i+1,j} + \delta_y a_{i,j}).\end{aligned}$$

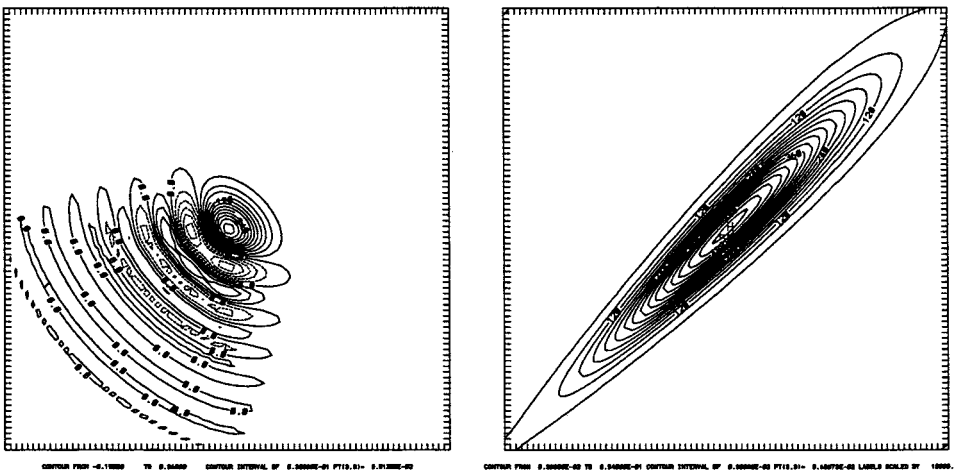

FIG. 8. Two-dimensional advection. The simulation in the left panel uses an improved Euler time discretization and centered spatial differences. The simulation in the right panel is the result of the exact enslaved scheme.

Similarly, the enslaving for the c variable is

$$c_{i,j+1} - c_{i,j} = -\frac{1}{8}\left(\delta_x^2 a_{i,j} + \delta_x^2 a_{i,j+1}\right) + \frac{C_1 \Delta x}{16\nu}(\delta_x a_{i,j+1} + \delta_x a_{i,j}) + \frac{C_2 \Delta x}{32\nu}(\delta_y^+ a_{i+1,j} + 6\delta_y^+ a_{i,j} + \delta_y^+ a_{i-1,j}).$$

The variable $d_{i,j}$ is not enslaved (i.e., is set to zero) since it represents a higher-order correction.

Figure 8 illustrates, as in the one-dimensional case, the positive definiteness of the enslaved scheme. A uniform tensor product mesh is used with $\Delta x = \Delta y = 1/80$, and the initial data is a Gaussian-shaped hill centered at the origin with height unity; specifically, $u(x, 0) = e^{-400(x^2+y^2)}$. In Fig. 8 we plot contours of the advected scalar $u(x, t)$ in (4.3) on a unit box with periodic boundary conditions. The time integration uses a second-order modified Euler discretization with $\Delta t = 0.001$. The velocity has components (1, 1), and the diffusion constant is $\nu = 0.0007$ giving a grid Reynolds number of approximately 18. The enslaved scheme is highly diffusive, but remains positive definite and monotone.

4.2. Shallow-Water Equations

As a final example we consider a more realistic geophysical application. The two-dimensional shallow-water equations are a rational approximation to the three-dimensional Euler equations, with the further assumption of a hydrostaticity. By including latitude-dependent wind and Coriolis forces, these equations describe a simple approximation to the depth-averaged dynamics of a large ocean basin. The equations are

$$\begin{aligned} &\frac{\partial h}{\partial t} + \frac{\partial (uh)}{\partial x} + \frac{\partial (vh)}{\partial y} = 0 \\ &\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} = -\frac{\partial h}{\partial x} + f(1+\beta y)v + \nu \Delta^2 u + F^u \\ &\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} = -\frac{\partial h}{\partial y} - f(1+\beta y)u + \nu \Delta^2 v + F^v, \end{aligned} \tag{4.6}$$

where the vector F represents the imposed wind forcing chosen here to be

$$F^u = \tau \cos(\pi y / L_y)$$
$$F^v = 0.$$

As the original differencing scheme we choose a positive-definite two-pass scheme (MP-DATA, [8], [9]) for the advection of the height field. The momentum field is solved using centered spatial discretizations and an explicit second-order time integrator. The equations are solved on a staggered grid, with the height field h located at cell centers and both components of the velocity field (u, v) located on the nodes (Arakawa B grid).

In analogy to (3.6), the approximate enslavings for the small scale components of the velocity field are

$$b_{i,j} = \frac{a_{i+1,j} - a_{i-1,j}}{8},$$
$$c_{i,j} = \frac{a_{i,j+1} - a_{i,j-1}}{8}.$$

Due to the staggering of the B grid, the height field requires a different change of variables (see [7]). The enslaving however is exactly the same.

In order to examine the validity and efficiency of the enslaving procedure over a range of flow conditions, we consider two separate sets of physical parameters and resolutions producing distinctly different temporal evolutions. The specific parameter values for the two cases are shown in Tables I and II. In each simulation, the flow was spun up from a zero velocity state under the action of the time independent wind forcing. Throughout, we will consider resulting differences in the domain averaged kinetic energy

$$E(t) = \int_\Omega \left[\frac{1}{2} h(x, y, t)(u^2(x, y, t) + v^2(x, y, t))\right] d\Omega,$$

as a measure.

TABLE I
Physical Parameters for the First Simulation

Coriolis parameter:		$f_0 = 5.0 \times 10^{-5}\ \mathrm{s}^{-1}$
$f = f_0(1 + \beta y)$		$\beta = 0.75$
Wind stress		$\tau_0 = 0.11\mathrm{N\ m}^{-2}$
Viscosity parameter		$\nu = 300\ \mathrm{m}^2\ \mathrm{s}^{-1}$
Effective gravity		$g' = 0.03\ \mathrm{ms}^{-2}$
Initial upper layer depth		$H_0 = 500$ m
Domain:		
	East–West	$L_x = 1000$ km
	North–South	$L_y = 2000$ km
Grid resolution:		
	Run S33	$\Delta x = 33$ km
	Run S16	$\Delta x = 16$ km
	Run E33	$\Delta x = 33$ km

TABLE II
Physical Parameters for the Second Simulation

Coriolis parameter:		$f_0 = 8.0 \times 10^{-5}$ s^{-1}
$f = f_0(1 + \beta y)$		$\beta = 0.4$
Wind stress		$\tau = 0.1$N m^{-2}
Viscosity parameter		$\nu = 225$ m^2 s^{-1}
Effective gravity		$g' = 0.03$ m s^{-2}
Initial upper layer depth		$H_0 = 500$ m
Domain		
	East–West	$L_x = 3000$ km
	North–South	$L_y = 2000$ km
Grid resolution:		
	Run SS50	$\Delta x = 40$ km
	Run SS100	$\Delta x = 20$ km
	Run EE50	$\Delta x = 40$ km

For the first set of physical parameters given in Table I and corresponding to a double-gyre situation previously studied by Jiang, Jin, and Ghil [2], the resulting flow is quasi-periodic in time. As shown in the first panel of Fig. 9, the underresolved ($\Delta x = 33$ km) standard scheme produces extraneously high kinetic energy and a marked low frequency oscillation. In contrast, the enslaved scheme run at the same spatial resolution produces a kinetic energy signal comparable in both amplitude and frequency to that of the standard scheme run at twice the resolution. Also shown are snapshots of the upper layer depth ($h(x, y, t)$) for the three schemes on a randomly chosen day in the eleventh year of the simulation. Again,

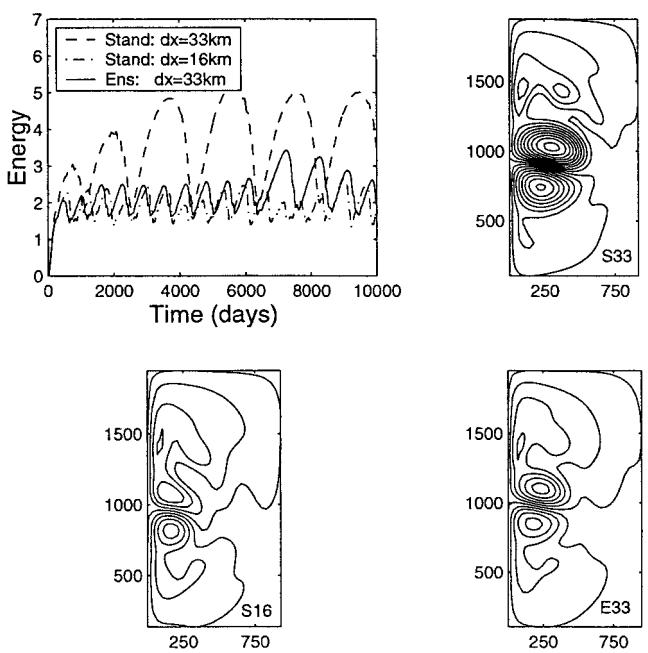

FIG. 9. Time trace of the averaged kinetic energy and contour plots of the upper layer depth, $h(x, y, t)$ for the three simulations.

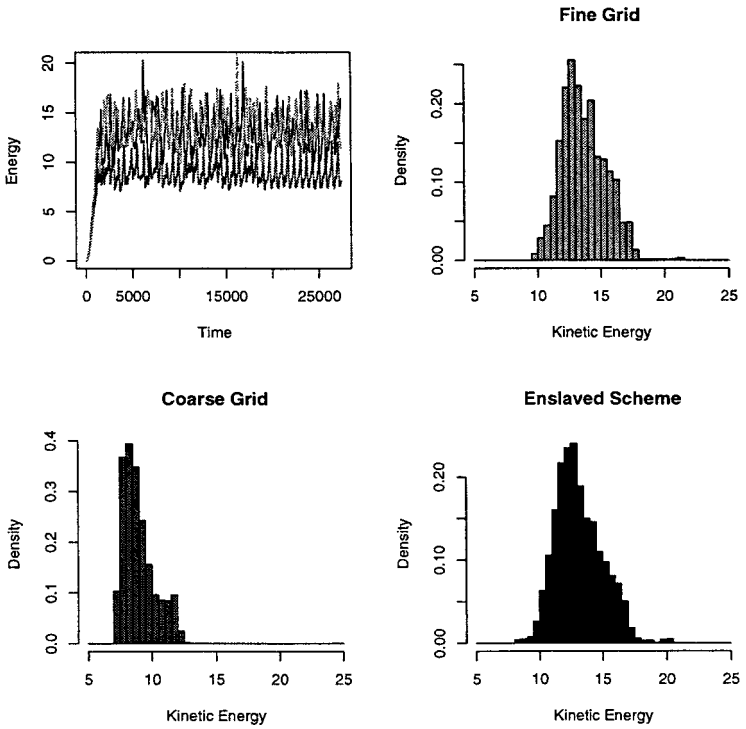

FIG. 10. Time trace and histograms of the averaged kinetic energy for the flow given by the parameters in Table II.

the enslaved scheme at the coarse, 33 km, resolution reproduces the smaller scale spatial structure seen in the fine resolution but absent in the coarse standard scheme simulation. It is the dynamics of these mesoscale eddies that determine the temporal behavior of the basin averaged energy.

The second set of physical parameters (given in Table II) produces a much higher Reynolds number flow with chaotic dynamics. In contrast to the quasi-periodic case, improved spatial resolution now leads to a large increase in the basin averaged kinetic energy signal. Figure 10 shows the time traces and histograms of the kinetic energy for the standard scheme at 20- and 40-km resolution and the enslaved scheme at the coarse resolution. As seen in the histograms, the enslaved scheme reproduces both the higher mean energy and the broader distribution of energy states produced by the fine grid standard scheme.

While we do not expect the enslaved scheme to track the instantaneous flow field of the more finely refined simulation in the chaotic regime, it is sufficient for it to capture the statistical features of the flow. Figure 11 shows the mean and the first three eigenvectors of the upper layer depth field correlation matrix (the first three empirical orthogonal functions) computed over a 60-year time span beginning at year 10 of the three simulations. In each, the enslaved scheme reproduces to a rather high degree of fidelity the statistical features given by the fine resolution standard scheme despite the large temporal variability in the solution. The dynamics of the enslaved scheme capture both the temporal and spatial modes that contribute to the variability of the flow.

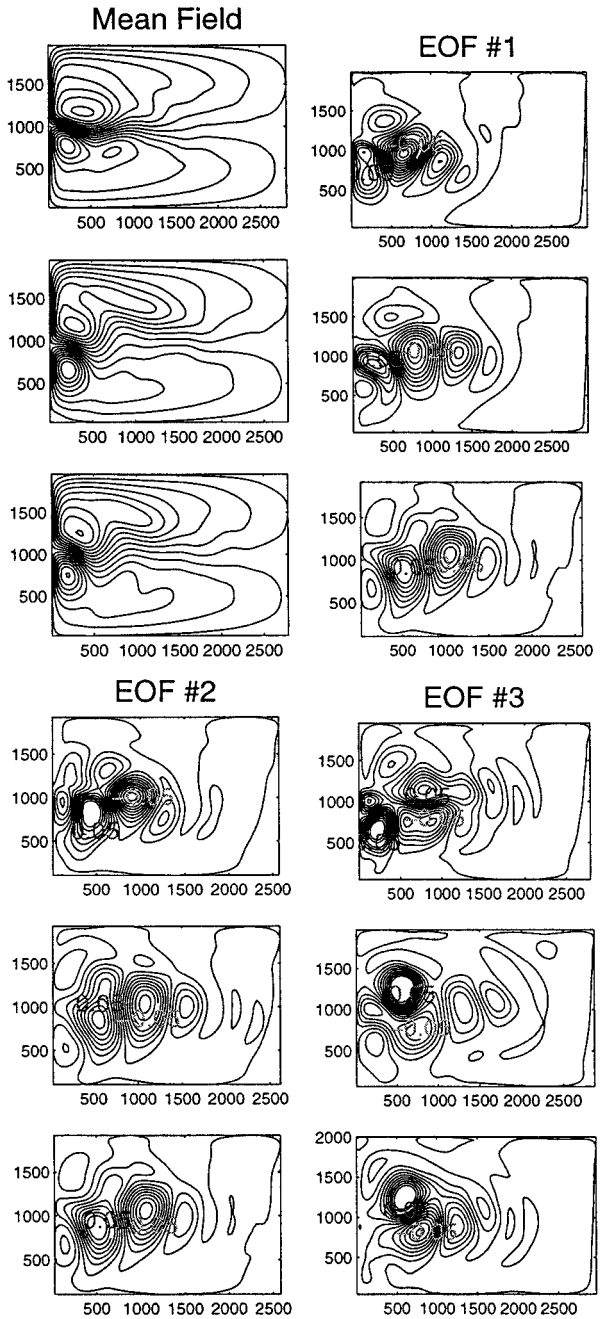

FIG. 11. Contours of the mean and first three EOFs of the upper layer depth for the chaotic simulations. In each, the standard scheme at 40 km resolution is at the top, the standard scheme at 20 km resolution is in the middle, and the enslaved scheme at 40 km resolution is shown at the bottom.

5. CONCLUSIONS

We have shown that use of enslaving, where the effects of unresolved scales of motion are estimated in terms of the resolved scales, can improve the accuracy, computational

efficiency, and fidelity of simulations in parameter regimes near steady state. Such parameter regimes, where the time derivatives are not part of the principal balance, are characteristic of many interesting flows in geophysics, reaction–diffusion kinetics, etc.

The methodology of enslaving does not construct an algorithm, but rather modifies a given algorithm. Specifically, the modified algorithm is constructed to reproduce the averaged results of a finely resolved simulation on a coarser mesh. We have described this construction procedure for finite-difference and finite-volume schemes in a general framework, which should be extensible to many evolution equations beyond the examples we have chosen.

We have described and compared two techniques of enslaving. The first is based on truncation analysis and ignores information that can be derived from the evolution equations themselves. The second enslaving results from analyzing asymptotic balances in the evolution equations of the small scales. The first enslaving is simpler, both conceptually and for practical implementation, and is more computationally efficient. The second enslaving has the additional feature that it produces nonoscillatory schemes even when the original schemes do not have this property.

Finally, we have described an application to a chaotic geophysical flow to illustrate that the enslaved schemes, run on coarse grids, can accurately reproduce the statistical properties of these flows as found on more finely resolved grids.

ACKNOWLEDGMENTS

The authors gratefully acknowledge the support of the Institute for Geophysics and Planetary Physics (IGPP) and the Center of Nonlinear Studies (CNLS) at Los Alamos National Laboratory, which is operated by the U.S. Department of Energy under Contract W-7405-ENG-36. ACP was also supported, in part, by PSC-CUNY Award 62809-00-31.

REFERENCES

1. A. Harten, High resolution schemes for hyperbolic conservation laws, *J. Comput. Phys.* **49**, 357 (1983).
2. S. Jiang, F.-F. Jin, and M. Ghil, Multiple equilibria, periodic and aperiodic solutions in a wind-driven, double-gyre, shallow water model, *J. Phys. Ocean.* **25**, 764 (1995).
3. D. A. Jones, L. G. Margolin, and A. C. Poje, Nonlinear difference approximations for evolutionary PDEs, in *Nonlinear Evolution Equations and Dynamical Systems*, *NEEDS*, edited by V. G. Makhankov, A. R. Bishop, and D. D. Holm (World Scientific, Singapore, 1994), pp. 65–75.
4. D. A. Jones, L. G. Margolin, and A. C. Poje, Enslaved finite difference schemes for nonlinear PDEs, *Numer. Meth. PDEs* **12**, 13 (1996).
5. D. A. Jones, A. C. Poje, and L. G. Margolin, Resolution effects and Enslaved finite difference schemes for a double gyre, shallow water model, *J. Theor. Comp. Fluid Dyn.* **9**, 269 (1997).
6. L. G. Margolin and D. A. Jones, An approximate inertial manifold for computing Burgers equation, *Physica* **D60**, 175 (1992).
7. A. C. Poje, D. A. Jones, and L. G. Margolin, Enslaved finite difference approximations for quasi-geostrophic shallow flows, *Physica* **D98**, 559 (1996).
8. P. K. Smolarkiewicz, A fully multidimensional positive definite advection transport algorithm with small implicit diffusion., *J. Comput. Phys.* **54**, 352 (1984).
9. P. K. Smolarkiewicz and L. G. Margolin, MPDATA: A finite-difference solver for geophysical flows, *J. Comput. Phys.* **140**, 459 (1998).
10. R. J. LeVeque, *Numerical Methods for Conservation Laws* (Birkhauser, Basel, 1992).