JOURNAL OF COMPUTATIONAL PHYSICS **123,** 466–475 (1996)
ARTICLE NO. 0037

Spline-Characteristic Method for Simulation of Convective Turbulence

ANDREI V. MALEVSKY

Université de Montréal, Département de Physique, et Centre de Recherche en Calcul Appliqué, 5160, boul. Décarie, bureau 400, Montréal, Québec H3X 2H9, Canada

Received June 25, 1993; revised June 13, 1995

This paper describes the spline-characteristic method, an algorithm for direct numerical simulation of incompressible, thermal convection in which the method of characteristics has been combined with tricubic splines. The method of characteristics eliminates errors associated with large gradients of convected field and allows large time steps, and the splines provide an accurate spatial approximation. The proposed scheme is second-order in time and fourth-order in space. The applications of the spline-characteristic method to the vorticity-transport equation and to the Navier–Stokes equations in primitive variables in the framework of the projection method are discussed. This scheme can be efficiently parallelized and has been implemented on a distributed-memory, massively parallel computer, CM-5. The results are given for several test problems with different boundary conditions, both in 2D and 3D. © 1996 Academic Press, Inc.

1. INTRODUCTION

When advection dominates diffusion, the advection–diffusion equation exhibits nearly hyperbolic behavior, and the numerical treatment of it is a challenging problem. The spectral, finite-difference, and finite-element schemes all have to have recourse to a small time step to maintain stability. A method of characteristics has been widely used for the treatement of various transport problems including fluid flow in porous media [1, 2] and weather simulations [3, 4]. The use of characteristics allowed us to overcome the Courant–Friedrichs–Lewy (CFL) time step limit in hyperbolic transport problems. The idea of using characteristics of the corresponding hyperbolic equation to discretize the total time derivative in a parabolic advection–diffusion equation had been put forward by Pironneau [5], Benqué *et al.* [6], and Douglas and Russell [7] more than a decade ago. Since then, the practical utilization of this idea to solve large-scale problems has been impeded by the difficulty to find an interpolation algorithm which is both efficient and accurate enough not to introduce significant aritificial smoothing of the solution.

The combination of the method of characteristics with the Galerkin approach is often referred as a modified method of characteristics [1] or a characteristic–Galerkin method [8]. In two-step characteristic (transport + diffusion) schemes [9, 10], the material is first transported and then an elliptic problem is solved. According to this approach, the advection–diffusion equation

$$\frac{\partial \theta}{\partial t} + \mathbf{u} \cdot \nabla \theta = \nabla^2 \theta \tag{1}$$

can be discretized with the finite elements as

$$\int_\Omega \theta_{i+1} N\, dV - \Delta t \int_\Omega \nabla^2 \theta_{i+1} N\, dV = \int_\Omega \theta_i(\mathbf{R}_{i+1}) N\, dV, \tag{2}$$

where $\mathbf{R}_{i+1}$ is the solution of a vector ODE

$$\frac{d\mathbf{R}}{dt} = \mathbf{u}(\mathbf{R}, t) \tag{3}$$

and θ_i is approximate solution of (1) at t_i; $\Delta t = t_{i+1} - t_i$ is a time step, and N is a test function. Equation (3) defines the characteristics of corresponding hyperbolic transport equation. This scheme is unconditionally stable [5] and conservative up to quadrature errors on the approximation of the integral in the right-hand side of (2) [8]. In practice, Eq. (1) can be nonlinear—the velocity may depend on θ. A predictor–corrector procedure was employed in [11] to approximate the unknown velocity in order to calculate the solution of (3). Calculation of the integral in the right-hand side of (2) requires interpolation to obtain the value of θ between the grid points; and the interpolation inevitably leads to numerical diffusion. The space-time, mixed finite element method with the elements oriented along characteristics has been recently proposed to eliminate numerical diffusion and ensure element-by-element conservation [12]. This method requires the use of deformable elements and, therefore, can be computationally expensive. Cubic splines have demonstrated excellent properties regarding numerical dissipation and phase errors in treatment of the advection–diffusion equation [13, 14]. This approach was employed in [11], where bicubic spline interpolation was used to keep the numerical diffusion low. The spline-characteristic scheme was applied to the heat-transport equation and tested on the 2D benchmark problems [11] which are the de facto standard for infinite Prandtl number convection [15, 16].

The extension of method of characteristics to the incompressible Navier–Stokes problem seems natural. Pironneau [5] proposed to use the divergence-free elements in the framework of the method of characteristics to model an incompressible flow. A fully implicit, finite-element scheme for the Navier–Stokes equation was outlined in [17]. Boukir *et al.* [10] have employed an Uzawa algorithm to solve the Stokes problem. The practical implementation of these ideas has been impeded by the numerical dissipation due to repeated interpolation. The spline-characteristic method has proven its ability to control the artificial diffusion [11], but the spline-basis for the characteristic–Galerkin method shared a serious disadvantage with the spectral schemes: it restricted geometry of a computational domain and the versatility was sacrificed for the sake of performance. The cubic splines have many valuable properties [18, 19] which facilitate their use for the spatial approximation of PDEs. A spline-collocation scheme was used in [20] to solve the 3D Navier–Stokes equations, but the spline–Galerkin approach yields higher accuracy.

The study of thermal convection has been motivated by industrial, astro- and geophysical applications. The discovery of transitions in the style of turbulent convection [21] facilitated numerical modeling of this phenomenon. Thermal convection combines two processes: the advection of momentum and the transport of heat. The relative importance of these two phenomena is defined by the Prandtl number (Pr). Thermal convection in a very viscous, high Pr fluid serves as a model of dynamical processes in the Earth's mantle; convection with low Pr is of interest to meteorology and astrophysics. The problems associated with the numerical treatment of the advective term in Navier–Stokes equations are well known. Convection with very high Prandtl and Rayleigh numbers (Ra) can hardly be reproduced in laboratory experiments, and numerical modeling remains the only feasible approach to this problem. In the case of convection with infinite Pr and high Ra—which simulates the mantle flow—the computational difficulties arise from vigorous advection of heat. This phenomenon has been numerically studied in 3D by means of the spectral methods [22, 23], and the spline-characteristic method [24]. In this paper, the spline-characteristic method is extended to a finite Pr convection; it is applied to both the Navier–Stokes equation and the thermal transport equation. We present below two formulations of this scheme, a vorticity-vector potential formulation and a pressure-correction formulation.

2. VORTICITY-VECTOR POTENTIAL FORMULATION

The Navier–Stokes equations for an incompressible flow can be reformulated in terms of the vorticity $\boldsymbol{\omega} = \nabla \times \mathbf{u}$, and the velocity can be obtained through a solenoidal vector potential defined by $\mathbf{u} = \nabla \times \mathbf{A}$ [25]. Then the Boussinesq approximation to thermal convection is described by the system

$$\begin{aligned} \frac{D\boldsymbol{\omega}}{Dt} &= -(\boldsymbol{\omega}\cdot\nabla)\mathbf{u} + \mathrm{Pr}\,\nabla^2\boldsymbol{\omega} + \mathrm{Pr}\cdot\mathrm{Ra}\cdot\nabla\times T\hat{z} \\ \frac{DT}{Dt} &= \nabla^2 T, \end{aligned} \tag{4}$$

where $D/Dt = \partial/\partial t + \mathbf{u}\cdot\nabla$ is the total (material) derivative, T is a temperature, $\mathbf{u}$ is a velocity, and $\hat{z}$ is a unit vector in the direction of gravity; the definitions for Ra and Pr can be found elsewhere (e.g., see [26]). Both equations in (4) are similar to the advection–diffusion equation (1), except for the vorticity-stretching term $(\boldsymbol{\omega}\cdot\nabla)\mathbf{u}$.

Let us choose a time step Δt, define $t_i = i\,\Delta t$, and consider an initial value problem $\mathbf{u}(t_0) = \mathbf{u}_0$, $T(t_0) = T_0$ for the system (4) on a rectangular prism $\Omega = [0, a] \times [0, b] \times [0, 1]$ with the grid $\mathbf{G} = (x_1, ..., x_{n_x}) \times (y_1, ..., y_{n_y}) \times (z_1, ..., z_{n_z})$ imposed on Ω. This restriction on a geometry of computational domain is essential to enable spline-interpolation (see the Appendix). Either Dirichlet or Neumann boundary conditions can be employed for T. The boundary conditions for vorticity and velocity will be discussed later. Let us assume that we already know an approximate solution of (4) for the time steps t_0, ..., t_i. The following scheme yields $\mathbf{u}_{i+1}$ and T_{i+1}, the approximate solution at $t = t_{i+1}$.

(a) We extrapolate the velocity using its values at the previous time steps to obtain an initial approximation to $\mathbf{u}(t_{i+1})$ (parabolic extrapolation can be used).

(b) We then solve a Cauchy problem for the ODE $d\mathbf{R}/dt = \mathbf{u}(\mathbf{R}, t)$ backwards in time from t_{i+1} to t_i using the extrapolated value of $\mathbf{u}(t_{i+1})$ obtained at the step (a). The nodes of spatial grid serve as the initial values for this problem $\mathbf{R}(t_{i+1}) = \mathbf{G}$. Thus the grid points are traced backwards by the characteristics to their previous positions $\mathbf{R}(t_i)$. A second-order Runge–Kutta scheme is used to solve the above ODE.

(c) Let us define $\mathbf{F}(\boldsymbol{\omega}, \mathbf{u}, T) = -(\boldsymbol{\omega}\cdot\nabla)\mathbf{u} + \mathrm{Pr}\cdot\mathrm{Ra}\cdot\nabla\times T\hat{z}$. At this stage, we need to calculate $\boldsymbol{\omega}(\mathbf{R}(t_i), t_i)$, $\mathbf{F}(\mathbf{R}(t_i), t_i)$, $T(\mathbf{R}(t_i), t_i)$. Note that the points $\mathbf{R}(t_i)$ do not generally coincide with the nodes $\mathbf{G}$. Therefore, this step requires an interpolation and introduces a certain amount of numerical diffusion. We use spline interpolation to calculate $\boldsymbol{\omega}(\mathbf{R}(t_i), t_i)$, $\mathbf{F}(\mathbf{R}(t_i), t_i)$, $T(\mathbf{R}(t_i), t_i)$ from the values of $\boldsymbol{\omega}(t_i)$, $\mathbf{u}(t_i)$, $T(t_i)$ at the grid points.

(d) We can now calculate vorticity $\boldsymbol{\omega}_{i+1}$ and temperature T_{i+1} by solving

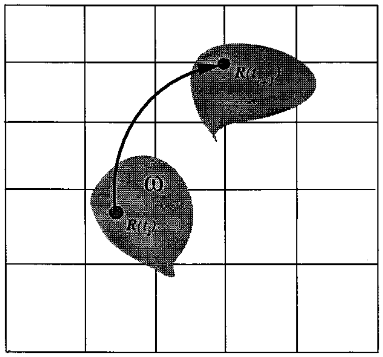

FIG. 1. Vorticity $\boldsymbol{\omega}$ is transported by the flow from its old position $\mathbf{R}(t_i) = (x(t_i), y(t_i), z(t_i))^T$ to the new location $\mathbf{R}(t_{i+1})$ along the characteristic. If $\mathbf{R}_{i+1}(t_{i+1})$ is a node of computational grid at the time t_{i+1}, then $\mathbf{R}_{i+1}(t_i)$ refers to its previous position at the time t_i. The vorticity-transport equation is discretized with respect to $\boldsymbol{\omega}_i(\mathbf{R}_{i+1}(t_i))$, and an interpolation is needed to calculate vorticity between the grid points.

$$\int_\Omega \boldsymbol{\omega}_{i+1} B\, dV - \mathrm{Pr}\,\Delta t \int_\Omega \nabla^2 \boldsymbol{\omega}_{i+1} B\, dV$$
$$= \int_\Omega \boldsymbol{\omega}(\mathbf{R}(t_i)) B\, dV + \Delta t \int_\Omega \mathbf{F}(\mathbf{R}(t_i)) B\, dV, \tag{5}$$

$$\int_\Omega T_{i+1} B\, dV - \Delta t \int_\Omega \nabla^2 T_{i+1} B\, dV = \int_\Omega T(\mathbf{R}(t_i)) B\, dV, \tag{6}$$

where B is a trial function, a basis tricubic spline in our case. In order to calculate integrals in the right-hand sides of (5) and (6), we assign the already known values of $\boldsymbol{\omega}(\mathbf{R}(t_i), t_i)$, $\mathbf{F}(\mathbf{R}(t_i), t_i)$, $T(\mathbf{R}(t_i), t_i)$ to the grid points and then calculate the spline representation of these fields. In other words, we project the grid values to a spline space by calculating a vector of coefficients $(c_1, ..., c_n)$ such that

$$\boldsymbol{\omega}(x_p, y_q, z_r) = \sum_{j=1}^{n} c_j B_j(x_p, y_q, z_r) \tag{7}$$

at every grid point (x_p, y_q, z_r), $p = 1, ..., n_x$; $q = 1, ..., n_y$; $r = 1, ..., n_z$ and $n = n_x n_y n_z$ is the total grid size. We also calculate the spline coefficients of $\mathbf{F}$ and T. A Galerkin form with splines employed in (5) and (6) leads to two systems of linear equations. We obtain the spline coefficients of ω_{i+1} and T_{i+1} by solving these systems.

(e) We can now find velocity $\mathbf{u}(t_{i+1})$ by solving three Poisson equations $\nabla^2 \mathbf{A}_{i+1} = -\boldsymbol{\omega}_{i+1}$ for the components of vector potential.

The ODE $d\mathbf{R}/dt = \mathbf{u}(\mathbf{R}, t)$ defines characteristic curves of a hyperbolic advection equation. We trace vorticity backwards to find $\boldsymbol{\omega}(\mathbf{R}(t_i), t_i)$, the vorticity at the time instant t_i at the point $\mathbf{R}(t_i) = (x(t_i), y(t_i), z(t_i))^T$. It is the point from where a vortex was transported to a node of computational grid along a characteristics curve (see Fig. 1). The steps (b)–(e) of the above algorithm can be repeated utilizing the newly found velocity $\mathbf{u}(t_{i+1})$ to solve the characteristic ODE. Thus a higher order time-stepping scheme can be constructed. The above spatial discretization is implicit with respect to the diffusive term and explicit for the vorticity-stretching and forcing terms. The vorticity-stretching term vanishes in a 2D flow, but it leads to the coupling of all the three vorticity components in the 3D case.

For this formulation, the boundary conditions are well known (e.g., see [27]) for two important cases: for a free-slip—also called a stress-free—boundary and for a rigid boundary. Let us assume $\mathbf{n}$ to be the unit vector normal to a bounding plane and $\mathbf{s}$ and $\boldsymbol{\tau}$ the orthogonal unit vectors laying within a bounding plane. Then the conditions on an impenetrable boundary of a rectangular prism Ω are

for the free-slip (zero tangential stress) boundary,

$$\omega_n = \omega_s = \omega_\tau = \partial A_n/\partial n = A_s = A_\tau = 0 \tag{9}$$

and for the rigid (zero velocity) boundary,

$$A_n = A_s = A_\tau = \partial A_n/\partial n = \partial A_s/\partial n = \partial A_\tau/\partial n = 0. \tag{10}$$

The free-slip boundary conditions (9) can be incorporated into the spline-basis for $\boldsymbol{\omega}$ and $\mathbf{A}$ as the substantial boundary conditions. For a rigid boundary, the $\boldsymbol{\omega} - \mathbf{A}$ system is inextricably coupled. Then (5) can be replaced by a fourth-order equation with respect to $\mathbf{A}$. Although this equation could be solved in a spline-space, it leads to an ill-conditioned matrix which is a serious obstacle for the iterative solvers.

The momentum equation in (4) is an advection–diffusion equation for vorticity with a forcing term (buoyancy) and yet with another additional term, the vorticity stretching term $(\boldsymbol{\omega} \cdot \nabla)\mathbf{u}$. When discretized with characteristics, an advection–diffusion equation does not contain the advective term and therefore allows time steps that are larger than the one dictated by the CFL criterion [5]. Discretization of the vorticity stretching term introduces a stability constraint on a time step. We are presently unable to obtain an exact formula for this constraint. In order to do that, we would need to estimate a norm of the operator $\boldsymbol{\omega} \cdot \mathbf{curl}^{-1}$. The vorticity stretching term is small at high Pr, and it is the viscous term which controls the time step. The discretization in (5) and (6) is implicit with respect to the diffusive terms $\mathrm{Pr} \cdot \nabla^2 \boldsymbol{\omega}$ and $\nabla^2 T$. An explicit discretization can also be used, provided that the time step is small enough to ensure stability.

3. VELOCITY-PRESSURE FORMULATION

The primitive variables, velocity $\mathbf{u}$ and pressure P, allow greater flexibility in the choice of boundary conditions for the system

$$\frac{D\mathbf{u}}{Dt} = \Pr \nabla^2\mathbf{u} - \nabla P + \Pr \cdot \mathrm{Ra} \cdot \left(\frac{1}{\alpha} - T\right) \hat{z} \tag{11}$$

$$\frac{DT}{Dt} = \nabla^2 T \tag{12}$$

$$\nabla \cdot \mathbf{u} = 0, \tag{13}$$

where α is the coefficient of thermal expansion. There are several approaches to ensure the incompressibility constraint (13). Pironneau [5] employed the divergence-free triangular elements to discretize the Navier–Stokes equations for the incompressible flow in a framework of the characteristic–Galerkin method. In practice, a higher order interpolation than linear is needed to keep the numerical diffusion low. The projection method [28, 29] can be used with both the finite elements and the finite differences of any order. The following semi-implicit algorithm combines the spline-characteristic scheme with the projection method:

(a) We first predict $\mathbf{u}(t_{i+1})$ by extrapolating velocities from the previous time steps.

(b) After that, we find the previous positions of the grid points (see Section 2) employing the predicted value of $\mathbf{u}(t_{i+1})$.

(c) We calculate values of velocity $\mathbf{u}_i(\mathbf{R}_{i+1}(t_i))$ and temperature $T_i(\mathbf{R}_{i+1}(t_i))$ at these locations using spline interpolation.

(d) We now assign these values to the nodes of computational grid and project them into the spline-space to calculate the right-hand sides for

$$\begin{aligned}\int_\Omega \tilde{\mathbf{u}} B\, dV - \Pr \Delta t \int_\Omega \nabla^2 \tilde{\mathbf{u}} B\, dV &= \int_\Omega \mathbf{u}_i(\mathbf{R}_{i+1}(t_i)) B\, dV + \mathrm{Ra} \cdot \Pr \cdot \int_\Omega \hat{z} B\, dV \\ &\quad - \mathrm{Ra} \cdot \Pr \cdot \int_\Omega \hat{z} T_i(\mathbf{R}_{i+1}(t_i)) B\, dV \end{aligned} \tag{14}$$

$$\int_\Omega T_{i+1} B\, dV - \Delta t \int_\Omega \nabla^2 T_{i+1} B\, dV = \int_\Omega T(\mathbf{R}(t_i)) B\, dV \tag{15}$$

as in Section 2.

(e) We solve (14) to get the intermediate velocity $\tilde{\mathbf{u}}$.

(f) We use the intermediate velocity to calculate the pressure from $\tilde{\mathbf{u}}$ as it is done in the projection scheme

$$\nabla^2 P = \frac{1}{\Delta t} \nabla \cdot \tilde{\mathbf{u}}. \tag{16}$$

(g) We finally correct the velocity $\mathbf{u}_{i+1} = \tilde{\mathbf{u}} - \Delta t\, \nabla P$.

This scheme requires boundary conditions for both the velocity and the pressure. The velocity boundary conditions are $u_n = \partial u_s/\partial s = \partial u_\tau/\partial \tau = 0$ for the free-slip boundary and $u_n = u_s = u_\tau = 0$ for the rigid boundary. The choice of boundary condition for the pressure is somewhat more complicated. Gresho [29] has shown that the proper boundary condition for pressure is the Neumann condition, obtained by applying the normal component of the momentum equation. The normal component of (11) on the boundary Γ yields

$$-\frac{\partial P}{\partial n} + \Pr \frac{\partial^2 u_n}{\partial n^2} + f_n = 0, \tag{17}$$

where f_n is a normal component of the gravity force $f = \Pr \cdot \mathrm{Ra} \cdot (1/\alpha - T)\hat{z}$. Since (13) holds everywhere on Ω, including Γ, then

$$\frac{\partial}{\partial n} \nabla \cdot \mathbf{u} = \frac{\partial^2 u_n}{\partial n^2} + \frac{\partial}{\partial s}\frac{\partial u_s}{\partial n} + \frac{\partial}{\partial \tau}\frac{\partial u_\tau}{\partial n} = 0. \tag{18}$$

Then on the free-slip boundary $\partial^2 u_n/\partial n^2 = 0$, and the boundary condition for pressure is $\partial P/\partial n = f_n$. If the isothermic boundary condition is imposed on Γ, i.e., $f = \mathrm{const}$, then the pressure boundary condition is decoupled from the velocity; and it can be incorporated into the spline-basis as a substantial boundary condition for the Poisson equation (16). For the rigid boundary $\partial^2 u_n/\partial n^2$ may be nonzero and the velocity and pressure are coupled by the boundary condition (17). Gresho [29] argued that a homogeneous Neumann boundary condition can be employed for pressure, but the discrepancy in the pressure boundary condition then introduces a spurious numerical boundary of the thickness $\delta = \sqrt{\Delta t}$, which must be small relative to any physical length scale, such as the thickness of thermal or viscous boundary layer in convection. A better approximation for the pressure boundary condition can be obtained by taking $\partial^2 u_n/\partial n^2$ from the preceding time step. Then the thickness of the spurious boundary layer would diminish to Δt.

The direct application of the spline–Galerkin scheme to (16) would lead to an underdetermined linear system, since the pressure is defined up to an arbitrary additive constant. The pressure could be "pinned" at one point, but it would cause poor convergence properties, and it is better to minimize simultaneously its average value

$$\int_\Omega \nabla^2 P B\, dV + \varepsilon \int_\Omega P B\, dV = -\frac{1}{\Delta t} \int_\Omega \nabla \cdot \tilde{\mathbf{u}} B\, dV \tag{19}$$

(ε is a small parameter), which is equivalent to the removal of the zeroth harmonic of pressure in a spectral scheme.

4. PARALLEL IMPLEMENTATION OF THE SPLINE-CHARACTERISTIC SCHEME

We describe here the parallel implementation of schemes from Sections 2 and 3. We will refer here to the scheme for primitive variables (Section 3) since both schemes share the same implementation details. The scheme has been implemented on a massively parallel, distributed-memory supercomputer, the CM5, in a single instruction multiple data (SIMD) mode. Programming in SIMD mode is usually less complex than in multiple instruction multiple data (MIMD) mode, where the data exchange between processors must be coded explicitly and a user must deal with the synchronization issues. Nevertheless the MIMD style programming ultimately grants better hardware utilization.

In the SIMD approach, each node of the spatial grid is associated with a virtual processor. A physical processor can accomodate several virtual processors; i.e., data from several grid points reside in the memory of a single physical processor. Operations involving data from different processors result in interprocessor communications which typically cost much more than the arithmetic operations within the memory of a single processor. The implementation strategy focuses on the reduction of interprocessor communications. Step (a) from Section 3, the extrapolation of the velocity using values from previous time steps, does not involve interprocessor communications since velocity is extrapolated at every grid point independently. Step (b) involves communications which follow unpredictable patterns. At this stage, we must find the previous locations of grid points along the characteristic curves (see Fig. 1). We use a second-order Runge–Kutta scheme to trace the grid points backwards. Thus, we first calculate the coordinates of intermediate locations $\tilde{\mathbf{R}} = \mathbf{R}(t_{i+1}) - \mathbf{u}_{i+1}\,\Delta t$, where $\mathbf{R}(t_{i+1}) = \mathbf{G}$ is the vector of coordinates of grid points. The coordinates of previous positions we seek are then given by $\mathbf{R}(t_i) = \mathbf{R}(t_{i+1}) - 0.5(\mathbf{u}_{i+1} + \mathbf{u}(\tilde{\mathbf{R}}))\,\Delta t$. We need to know the velocity in the intermediate locations. These locations can be elsewhere within the rectangular prism Ω and they do not generally coincide with the grid points. The velocity at the intermediate locations can be obtained by spline-interpolation of the velocity field $\mathbf{u}_i$ given by its spline representation. We only need to know the grid intervals where the points $\tilde{R}_{pqr} = (\tilde{x}_p, \tilde{y}_q, \tilde{z}_r)$ reside. A search procedure has to be implemented if the grid is nonequispaced. Since we assume a tensor product grid, the search can be conducted consequently for every spatial dimension and in parallel for every point. Thus we first seek the intervals on the x-axis for every point, then on the y-axis and the z-axis. It is reasonable to begin the search from the interval containing the starting grid point. First, we check whether $\tilde{x}_p$ belongs to the interval (x_p, x_{p+1}). If yes, we mask this point out and continue the search for the other points. We check (x_{p-1}, x_p), (x_{p+1}, x_{p+2}), etc., masking out the points which have been located until the intervals for all the $\tilde{R}_{pqr}$ are found. It is done in parallel for every point or for every triplet of indices p, q, r. We apply the same procedure to find the grid intervals along the y- and z-axes.

The next step (step (d) in Section 3) involves projection from the physical space to spline space. In order to find coefficients $(c_1, ..., c_n)$ of spline-representation (see (7) in Section 2), we need to solve a large number of tridiagonal systems. Due to the tensor-product nature of spline-interpolation, we can first project x-coordinates to 1D spline space and then y- and x-coordinates, consequently, the same way a multidimensional Fourier transform is done [19]. This operation is highly efficient on a parallel computer since all the tridiagonal systems are independent and can be solved simultaneously. We usually keep one of the axes local to a processor. We utilize a direct tridiagonal solver for this local axis and use an iterative solver for the two remaining nonlocal axes. We could, instead, make every axis local, in turn, by performing matrix transpose and thus solve only completely local tridiagonal systems. However, the transpose takes a noticable time due to the interprocessor communications involved and requires some temporary arrays. The multiple instance tridiagonal solvers themselves run very fast but some computer time and memory is lost by doing a transpose. We found that it is faster to use a combination of direct and iterative solvers which requires less communication and avoid the matrix transpose.

The Poisson solver is the most computationally intensive part of the scheme. The spline–Galerkin form of a pressure Poisson equation (19) results in a block-diagonal matrix. The matrices for Eqs. (5), (6), (15), and (16) have the same structure and properties. All the entries of these matrices are integrals of products of basis splines and their derivatives which are piecewise polynomials. The entries can be precomputed and stored in a table. The matrices are symmetric positive definite. We have employed a conjugate gradient iterative scheme to solve these systems. A matrix-by-vector multiplication controls the performance of the conjugate gradient solver. It is a regular, stencil-type matrix-by-vector multiplication and involves a certain amount of communication in a form of vector shifts. The other basis operation of a conjugate gradient solver, a dot product of two vectors, takes much less time than the matrix-by-vector product. A dot product belongs to a family of operations known as global reductions. The global reductions on the CM5 utilize a special low-bandwidth and low-latency network. In general, the spline-characteristic scheme can be efficiently parallelized.

5. TEST RESULTS

The chaotic behavior of a finite-Prandtl-number, 3D thermal convection has been examined numerically in sev-

TABLE I

The Benchmark Results and Their Relative Deviation δ from Those in [28]

	10^4		10^5		10^6	
Ra	Value	δ	Value	δ	Value	δ
Nu	2.219	−0.010	4.370	−0.033	8.000	−0.090
w max	19.59	−0.001	68.40	−0.003	218.5	−0.004
x	0.120	+0.001	0.068	+0.002	0.043	+0.005
u max	16.13	−0.003	35.60	+0.025	78.79	+0.219
z	0.824	+0.001	0.859	+0.004	0.885	+0.035
Nu max	3.468	−0.017	7.245	−0.061	14.67	−0.182
z	0.149	+0.006	0.096	+0.015	0.052	+0.014
Nu min	0.596	+0.017	0.790	+0.083	1.250	+0.262
z	1.000	+0.000	1.000	+0.000	1.000	+0.000

eral publications (e.g., [30, 31]). Even for the simpler cases, the numerical treatment of equations sometimes is fraught with blemishes—some physical phenomena may be suppressed by the inadequate resolution, or the numerical methods can introduce the spurious, nonphysical processes. The spatial resolution of numerical experiments can hardly be judged by simple examination of the energy contents of the Fourier modes, since some methods can introduce the artificial damping at high wavenumbers. Errors in the spatial approximation affect the temporal accuracy. Curry *et al.* [30] have demonstrated how the inadequate spatial resolution in numerical convection leads to the strong attenuation of high frequencies in the temporal domain. Therefore, it is always desirable to test the behavior and accuracy of algorithms experimentally and verify them against the existing benchmarks.

The 2D, steady-state benchmark for a finite-Prandtl-number convection [32] provides a basis for the limited assessment of the proposed algorithm. This test problem simulates convection with $\mathrm{Pr} = 0.71$ in the aspect ratio 1 box with all the rigid boundaries, and with z vertically upwards, the boundaries $z = 0$ and $z = 1$ assumed insulated, and $T = 1$ at $x = 0$ and $T = 1$ at $x = 1$. The same code, which was used for the time-dependent problems, was employed to get the steady-state solutions of this model problem. The spline-characteristic method with projection was applied to the system (11)–(13) in primitive variables (see Section 3) on the equispaced grid of 32×32 bicubic splines. The test results are summarized in Table I, where the calculated values are given, together with their relative deviation δ from the best solution in [32]. The goal was to test the real code, not to obtain the best approximation to the benchmark problem; and the results in Table I exhibit generally good matching with [32], except for $\mathrm{Ra} = 10^6$—the 32×32 grid insufficiently resolves the flow for this Ra. A better spatial resolution (64×64) yielded a satisfactory result for $\mathrm{Ra} = 10^6$ with the results deviating by not more than 1% of the values reported in [32].

A steady-state, 2D benchmark is clearly restricted in the evaluation of the virtues of the code intended to simulate the 3D turbulence. Therefore, the comparative 3D simulations have been done for a variety of Ra and Pr. Both formulations, the vorticity-vector potential and the velocity-pressure, were employed to model convection in the aspect-ratio $2\sqrt{2} \times 2\sqrt{2} \times 1$ box with the periodic boundary conditions on the vertical boundaries, and the free-slip conditions on the isothermic horizontal planes. Figure 2 exhibits the time series of the surface Nu and the root mean squared (RMS) velocity of convection with $\mathrm{Pr} = 1.0$ and $\mathrm{Ra} = 10^4$. A steady-state solution in the form of two counterrotating rolls was obtained after the initial transient stage. Both formulations yielded almost identical Nu on the same computational grid, but the RMS velocity was slightly less for the primitive variables. Although the initial conditions were different and the solutions had not yet converged completely to the steady state. The "safe" solution was calculated on a vertically nonequispaced grid of $64 \times 64 \times 32$ splines. The solution obtained on the vertically nonequispaced grid of $32 \times 32 \times 16$ splines was closer to this "safe" solution than one calculated on the equispaced grid. Another case, $\mathrm{Pr} = 10.0$ and $\mathrm{Ra} = 2.63 \times 10^4$, suitable for benchmarking, was provided in [30], where a single period solution was established after the transient stage. Both formulations of the spline-characteristic method yielded the periodic solution—two oscillating rolls—with the frequency of 43 (Fig. 3). This value is close to 39 reported in [30] taking into account a rather large margin of error in our case.

The above tests, although being calculated in the 3D geometry, gave solutions in a form of 2D rolls infinitely extended in the direction of a horizontal axis; and the vorticity-stretching term vanished in these, essentially 2D, cases. Increasing the Rayleigh number a fully 3D, chaotic solution was obtained for $\mathrm{Pr} = 1.0$ and $\mathrm{Ra} = 3.0 \times 10^5$ on the grid of $64 \times 64 \times 32$ splines using the vorticity-vector potential formulation, but this scheme failed to converge for $\mathrm{Pr} = 0.7$ and $\mathrm{Ra} = 6.5 \times 10^6$ on the $128 \times 128 \times 64$ grid. The explicit treatment of the vorticity-stretching in (5) caused the fiasco. A higher-order approximation was obviously needed for this term; and how to construct this approximation in the framework of spline-characteristic method is still an open question. The parameters for this case were taken from [33]. This simulation was described in detail and therefore provided an opportunity to test the proposed numerical scheme in the hard-turbulent regime. The spline-characteristic method with primitive variables (see Section 3) yielded a stable solution (Fig. 4). It is impossible to obtain exactly the same picture of a turbulent flow at each time instant with two different methods, but the main properties and the averaged quantities should match.

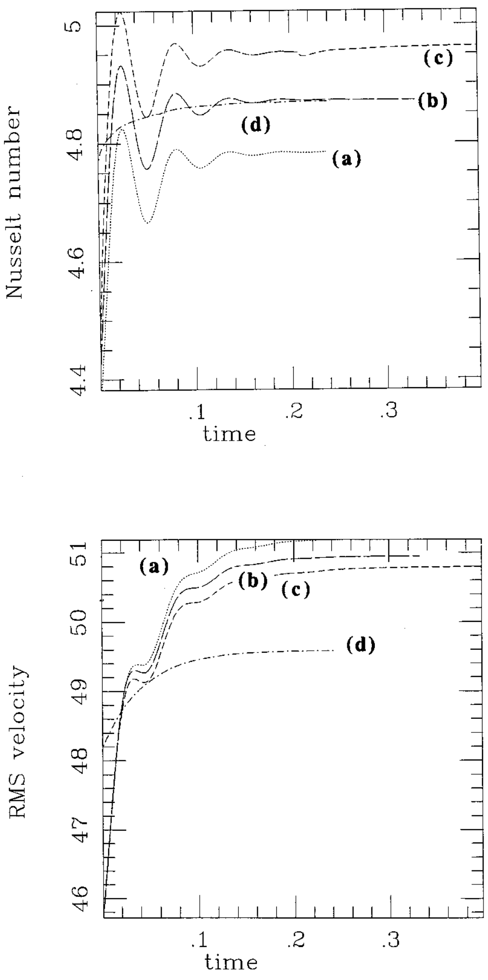

FIG. 2. The time series of the surface Nu and the RMS velocity of convection with Pr = 1.0 and Ra = 10^4 in a 3D box with the aspect ratio $2\sqrt{2} \times 2\sqrt{2} \times 1$, the free-slip horizontal boundaries and the periodic vertical boundaries. The temperature 0 was kept on the upper surface, and 1 on the lower surface. The time scale is given in the thermal diffusion units. The spline-characteristic method was applied to the vorticity-transport equation in (a) (...), (b) (——), and (c) (– – –), and to the Navier–Stokes equations in primitive variables employing the projection scheme in (d) (–·–·–). The initial conditions for (a)–(c) differ from those taken for (d). All cases yielded a steady-state solution, the two counterrotating rolls. A vertically nonequispaced grid with 32 × 32 × 16 cubic splines was used in (b) and (d); an equispaced grid with 32 × 32 × 16 splines was employed in (a); and (c) was calculated on a vertically nonequispaced grid of 64 × 64 × 32 splines.

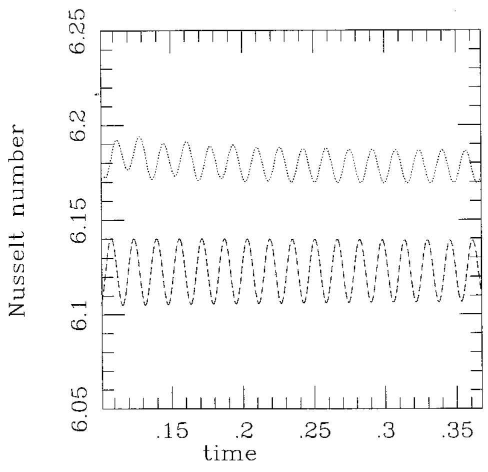

FIG. 3. The time series of the surface Nu of convection with Pr = 10.0 and Ra = 2.63×10^4 in a 3D box with the aspect ratio $2\sqrt{2} \times 2\sqrt{2} \times 1$, the free-slip horizontal boundaries and the periodic vertical boundaries. The temperature 0 was kept on the upper surface, and 1 on the lower surface. A vertically nonequispaced grid of 32 × 32 × 16 splines was employed. The time scale is given in the thermal diffusion units. Both the vorticity-vector potential (– – –) and the velocity-pressure (...) formulations yielded the periodic solution with a single frequency of approximately 43 after the initial transient stage.

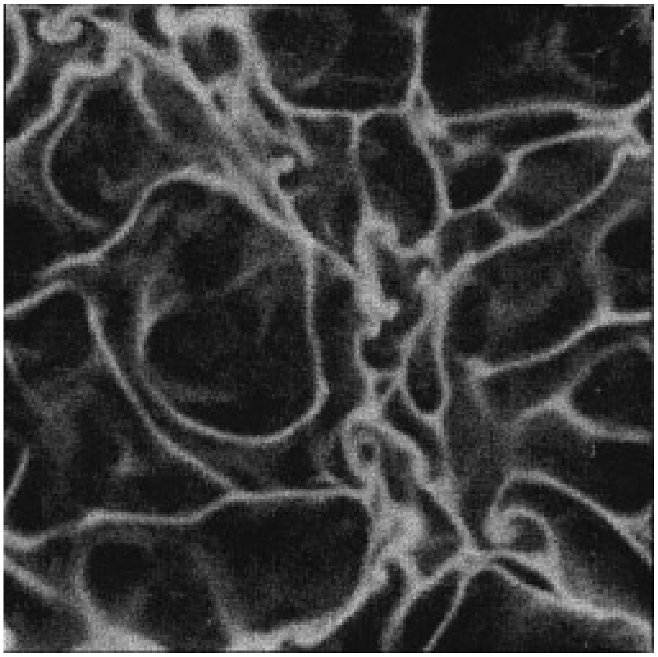

FIG. 4. A top view of convection with Pr = 0.7 (air) and Ra = 6.5×10^6 in a 3D box with the aspect ratio $2\sqrt{2} \times 2\sqrt{2} \times 1$, the free-slip horizontal boundaries and the periodic vertical boundaries—all the parameters are identical to those in [33]. This case—well described in the publications—has been recomputed to verify the algorithm in the hard-turbulent regime. The spine-characteristic + projection method was used on a grid of 128 × 128 × 64 cubic splines. The white "spaghetti" depict thermal structures in the cold boundary layer, the Benard cells.

All the quantities measured in our simulation agree well with the findings in [33]. We find the time-averaged Nu = 23.0 ± 0.6 versus 23.7 reported in [33]. The probability distribution functions (PDF) give an insight into the small scales of motion. The PDFs of the temperature and velocity fluctuations (Fig. 5) are similar to those in [33], although our statistics are based on significantly fewer realizations. The data for the PDFs are ensembled over a horizontal plane for several time instants. The power spectrum of

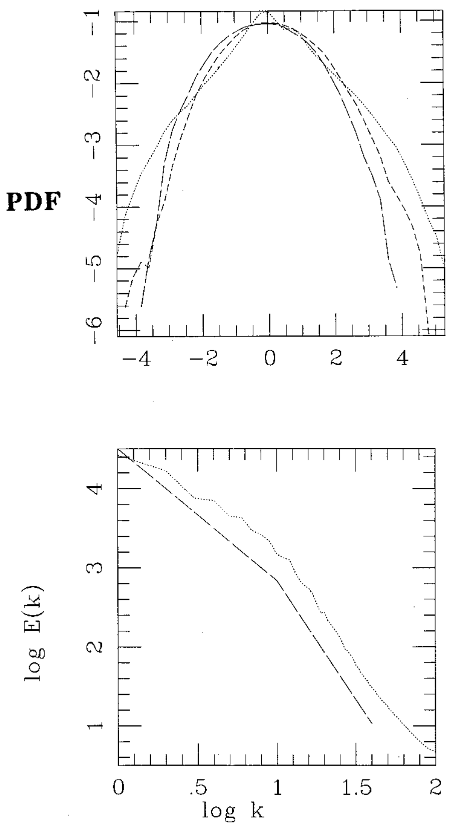

FIG. 5. Probability distribution functions (PDF) of the temperature fluctuations (...), fluctuations of one of the horizontal velocities (——), and the vertical velocity fluctuations (– – –) as they are defined in [29]; and the time-averaged spectrum of kinetic energy with $k^{-5/3}$ and k^{-3} shown for the reference (b). The PDFs and the spectrum are measured in the midplane $z = 0.5$.

velocity fluctuations in Fig. 5 is calculated by averaging both in time and over the polar angle in the horizontal plane. The spectrum exhibits two subranges and is similar to one in [34].

6. CONCLUSIONS

The spline-characteristic method combines the efficient treatment of advection with high-order spatial approximation. This method had previously proven its robustness in the numerical treatment of the heat-transport equation where advection dominates diffusion both in 2D [11] and in 3D [24]. It has been also applied to simulate low Pr convection in turbulent regime [35].

Both the vorticity-vector potential and velocity-pressure formulations are feasible, but the applicability of the vorticity-vector potential formulation is restricted, and the scheme may be unstable at high Re, due to the explicit handling of the vorticity-stretching term. A scheme implicit with respect to this term would couple the equations for velocity components. An extrapolation, e.g., Adams–Bashforth, scheme may resolve this problem, but the values of **F** in (5) at the previous time steps must be taken from the characteristic curves, not at the nodal points. The spline-characteristic method, combined with the projection scheme for the Navier–Stokes equations in primitive variables, maintains stability even at high Re. However the velocity-pressure scheme requires a very small time step if both Ra and Pr are high and the vorticity-vector potential formulation works better. The benchmark comparisons corroborate the accuracy of the proposed numerical scheme.

A significant number of numerical simulations have been conducted recently for the 3D thermal convection, sometimes addressing such delicate issues as the behavior of small scales in a turbulent regime. These developments have created prerequisites to build the rigorous 3D benchmark for the convection codes.

APPENDIX: TRICUBIC SPLINES

The cubic splines are well suited for the interpolation and the spatial approximation of PDEs. A multidimensional cubic spline consists of the polynomials of third order with respect to each spatial variable. These polynomials are welded at the nodal points to preserve the continuity of the function and its derivatives. A 3D cubic spline has the continuous derivatives $\partial^{i+j+k}/\partial x^i\ \partial y^j\ \partial\ \partial z^k$ for $i \leq 2, j \leq 2, k \leq 2$.

There is usually a distinction between an order of spline and an order of the polynomial which generates the spline in the spline literature. A polynomial of the order $m - 1$ generates a spline of the order m. Thus a cubic spline is said to have a fourth order. A 1D polynomial spline of order m is a piecewise polynomial, which can be expressed as a linear combination of the basis splines B with the local support over m grid intervals [19]. The basis splines B_j of order m can be defined as translations of the cardinal B-spline N_m: $B_j(x) = N_m(x - j)$. The mth order cardinal B-spline can be calculated recursively as a convolution $N_m = N_{m-1} * N_1$, or alternatively, as $N_m(x) = (xN_{m-1}(x) + (m - x)N_{m-1}(x - 1))/(m - 1)$, where $N_1(x) = \chi_{[0,1]}(x)$ is a box function [36]. The translations of N_m generates a basis in the space of splines of order m with periodic boundary conditions. The basis splines at the ends of the interval must be modified in order to represent another boundary condition.

The space of multidimensional splines can be expressed as a tensor product of the 1D spline spaces, or, in other words, each 3D basis spline is a product of three 1D basis splines. We use here a modified spline-basis. For the easy treatment of boundary conditions it is convenient to specify the derivatives of the B-splines adjacent to the boundary

TABLE II

The Spline-Basis for the Equispaced Grid $x_1 = 0$, $x_2 = h$, ..., $x_n = (n-1)h$; $\xi = (x - x_j)/h$

Spline/interval	$0 \le x \le h$	$h \le x \le 2h$	$2h \le x \le 3h$
$B_\alpha(\xi)$	$\frac{1}{2}\xi^2 - \frac{11}{36}\xi^3$	$\frac{7}{36} + \frac{1}{12}\xi - \frac{5}{12}\xi^2 + \frac{7}{36}\xi^3$	$\frac{1}{18} - \frac{1}{6}\xi + \frac{1}{6}\xi^2 - \frac{1}{18}\xi^3$
$B_\beta(\xi)$	$\xi - \frac{1}{3}\xi^3$	$\frac{2}{3} - \xi^2 + \frac{1}{2}\xi^3$	$\frac{1}{6} - \frac{1}{2}\xi + \frac{1}{2}\xi^2 - \frac{1}{6}\xi^3$
$B_\delta(\xi)$	$1 - \frac{1}{6}\xi^3$	$\frac{5}{6} - \frac{1}{2}\xi - \frac{1}{2}\xi^2 + \frac{1}{3}\xi^3$	$\frac{1}{6} - \frac{1}{2}\xi + \frac{1}{2}\xi^2 - \frac{1}{6}\xi^3$

$B_\gamma(\xi)$ for $j = 3, ..., n-2$

$(j-3)h \le x \le (j-2)h$	$(j-2)h \le x \le (j-1)h$	$(j-1)h \le x \le jh$	$jh \le x \le (j+1)h$
$\frac{1}{4}\xi^3$	$\frac{1}{4} + \frac{3}{4}\xi + \frac{3}{4}\xi^2 - \frac{3}{4}\xi^3$	$1 - \frac{3}{2}\xi^2 + \frac{3}{4}\xi^3$	$\frac{1}{4} - \frac{3}{4}\xi + \frac{3}{4}\xi^2 - \frac{1}{4}\xi^3$

	$(n-4)h \le x \le (n-3)h$	$(n-3)h \le x \le (n-2)h$	$(n-2)h \le x \le (n-1)h$
$\hat{B}_\alpha(\xi)$	$\frac{1}{18}\xi^3$	$\frac{1}{18} + \frac{1}{6}\xi + \frac{1}{6}\xi^2 - \frac{7}{36}\xi^3$	$\frac{7}{36} - \frac{1}{12}\xi - \frac{5}{12}\xi^2 + \frac{11}{36}\xi^3$
$\hat{B}_\beta(\xi)$	$\frac{1}{6}\xi^3$	$\frac{1}{6} + \frac{1}{2}\xi + \frac{1}{2}\xi^2 - \frac{1}{2}\xi^3$	$\frac{2}{3} - \xi^2 + \frac{1}{3}\xi^3$
$\hat{B}_\delta(\xi)$	$\frac{1}{6}\xi^3$	$\frac{1}{6} + \frac{1}{2}\xi + \frac{1}{2}\xi^2 - \frac{1}{3}\xi^3$	$\frac{5}{6} + \frac{1}{2}\xi - \frac{1}{2}\xi^2 + \frac{1}{6}\xi^3$

of a computational domain. Indeed each of the sets B_α, B_β, B_γ, B_γ, ...; B_δ, B_β, B_γ, B_γ, ...; B_δ, B_α, B_γ, B_γ, ... yields a unique B-spline basis on a 1D grid $x_1, ..., x_n$, where the piecewise cubic polynomials B_α, B_β, B_γ, B_δ are defined by the conditions:

$$\begin{aligned}
&B_\alpha(x_1) = 0, \quad B'_\alpha(x_1) = 1, \quad B''_\alpha(x_1) = 0,\\
&B_\alpha(x_4) = B'_\alpha(x_4) = B''_\alpha(x_4) = 0\\
&B_\beta(x_1) = 0, \quad B'_\beta(x_1) = 0, \quad B''_\beta(x_1) = 1,\\
&B_\beta(x_4) = B'_\beta(x_4) = B''_\beta(x_4) = 0\\
&B_\delta(x_1) = 1, \quad B'_\delta(x_1) = 0, \quad B''_\delta(x_1) = 0,\\
&B_\delta(x_4) = B'_\delta(x_4) = B''_\delta(x_4) = 0\\
&B_\gamma(x_1) = B'_\gamma(x_1) = B''_\gamma(x_1) = 0, \quad B_\gamma(x_3) = 1,\\
&B_\gamma(x_5) = B'_\gamma(x_5) = B''_\gamma(x_5) = 0.
\end{aligned} \tag{20}$$

The basis splines on the other end of the interval (x_1, x_n) can be designed in the same way.

The basis B_α, B_β, B_γ, B_γ, ... imposes the homogeneous Dirichlet boundary condition at x_1; the set B_δ, B_β, B_γ, B_γ, ... gives the homogeneous Neumann boundary condition; and B_δ, B_α, B_γ, B_γ, ... yields the second-order condition $f''(x_1) = 0$. The periodic boundary condition can be set up by taking n basis splines B_γ. The coefficients of the piecewise cubic polynomials B_j, $j = 1, ..., n$, are computed once for each grid by solving n linear systems of size 16×16 which define the four coefficients of the four polynomials composing a basis spline. These systems can be constructed by taking (20), together with the continuity conditions for the $B^{(k)}$, $k = 0, 1, 2$ at the nodal points. Coefficients of the polynomials which constitute the basis splines for an equispaced 1D grid are given in Table II, where $\xi = (x - x_j)/h$ denotes the local coordinate with respect to a nodal point x_j. This choice of basis makes the scheme flexible in the treatment of substantial boundary conditions.

ACKNOWLEDGMENTS

This research was supported by the Army High Performance Computer Research Center under ARO contract number DAAL03-89-C-0038. The suggestions of D. Eyre and A. Vincent are very much appreciated.

REFERENCES

1. R. E. Ewing, T. F. Russell, and M. F. Wheeler, *Comput. Methods Appl. Mech. Eng.* **47,** 73 (1984).
2. P. J. Roache, *Int. J. Numer. Methods Fluids* **15,** 1259 (1992).
3. A. Robert, *J. Meteor. Soc. Japan* **60,** 319 (1982).
4. A. Staniforth and J. Côté, *Mon. Weather Rev.* **119,** 2206 (1991).
5. O. Pironneau, *Numer. Math* **38,** 309 (1982).
6. J. P. Benqué, B. Ibler, A. Keramsi, and G. Labadie, ''A New Finite-Element Method For Navier–Stokes Equations Coupled with a Temperature Equation,'' in *Proceedings, 4th Int. Symp. on Finite Elements*

in Flow Problems, edited by T. Kawai (North Holland, Amsterdam, 1982), p. 295.

7. J. Douglas and T. F. Russell, *SIAM. J. Numer. Anal.* **19,** 871 (1982).
8. O. Pironneau, J. Liou, and T. Tezduyar, *Comput. Methods Appl. Mech. Eng.* **100,** 117 (1992).
9. C. Johnson, *Comput. Methods Appl. Mech. Eng.* **100,** 45 (1992).
10. K. Boukir, Y. Maday, and B. Métivet, *Comput. Methods Appl. Mech. Engrg.* **116,** 211 (1994).
11. A. V. Malevsky and D. A. Yuen, *Phys. Fluids A* **3,** 2105 (1991).
12. M. F. Wheeler and T. Arbogast, *SIAM J. Numer. Anal.,* submitted.
13. J. Pudykiewicz and A. Staniforth, *Atmos-Ocean* **22,** 283 (1984).
14. R. Bermejo, *Mon. Weather Rev.* **118,** 979 (1990).
15. B. Blankenbach, F. Busse, U. Christensen, L. Cserepes, U. Hansen, H. Harder, G. Jarvis, M. Koch, G. Marquardt, D. Moore, P. L. Olson, H. Schmeling, and T. Schaubelt, *Geophys. J.* **98,** 23 (1989).
16. B. J. Travis, C. Anderson, J. Baumgardner, C. W. Gable, B. H. Hager, R. J. O'Connell, P. Olson, A. Raefsky, and G. Schubert, *Geophys. Astr. Fluid Dyn.* **55,** 137 (1990).
17. M. O. Bristeau, R. Glowinski, and J. Periaux, "Numerical Methods for Navier–Stokes Equations. Applications to the Solution of Compressible and Incompressible Viscous Flow," in *Finite Elements in Physics,* edited by R. Gruber (North-Holland, Amsterdam, 1987), p. 73.
18. J. H. Ahlberg, E. N. Nilson, and J. L. Walsh, *The Theory of Splines and Their Applications* (Academic Press, London, 1967).
19. C. De Boor, *A Practical Guide to Splines* (Springer-Verlag, New York, 1978).
20. A. Nordlund and R. F. Stein, *Comput. Phys. Commun.* **59,** 119 (1990).
21. F. Heslot, B. Castaing, and A. Libchaber, *Phys. Rev. A* **36,** 5870 (1987).
22. S. Honda, D. A. Yuen, S. Balachandar, and D. Reuteler, *Science* **259,** 1308 (1993).
23. P. J. Tackley, D. J. Stevenson, G. A. Glatzmaier, and G. Schubert, *Nature* **361,** 699 (1993).
24. A. V. Malevsky and D. A. Yuen, *Geophys. Res. Lett.* **20,** 383 (1993).
25. C. A. J. Fletcher, *Computational Techniques for Fluid Dynamics II* (Springer-Verlag, New York, 1988), p. 338.
26. S. Chandrasekhar, *Hydrodynamic and Hydromagnetic Stability* (Dover, New York, 1961).
27. G. A. Houseman, *Geophys. J. Int.* **100,** 33 (1990).
28. A. J. Chorin, *Math. Comput.* **22,** 745 (1968).
29. P. M. Gresho, *Comput. Methods Appl. Mech. Eng.* **87,** 201 (1991).
30. J. H. Curry, J. R. Herring, J. Loncaric, and S. A. Orszag, *J. Fluid Mech.* **147,** 1 (1984).
31. M. Meneguzzi, C. Sulem, P. L. Sulem, and O. Thual, *J. Fluid Mech.* **182,** 169 (1987).
32. C. De Vahl Davies and I. P. Jones, *Int. J. Numer. Methods Fluids* **3,** 227 (1983).
33. S. Balachandar and L. Sirovich, *Phys. Fluids A* **3,** 919 (1991).
34. R. Kerr, *J. Fluid Mech.,* in press.
35. A. V. Malevsky, *Phys. Earth Planet. Int.* **88,** 31 (1995).
36. C. K. Chui, *An Introduction to Wavelets* (Academic Press, Boston, 1992).