JOURNAL OF COMPUTATIONAL PHYSICS **143,** 125–158 (1998)
ARTICLE NO. CP985956

# A Staggered-Grid Multidomain Spectral Method for the Compressible Navier–Stokes Equations

David A. Kopriva

*Supercomputer Computations Research Institute and Department of Mathematics, The Florida State University, Tallahassee, Florida 32306*
E-mail: dak@scri.fsu.edu

Received July 31, 1997; revised February 3, 1998

We present a flexible, non-conforming staggered-grid Chebyshev spectral multidomain method for the solution of the compressible Navier–Stokes equations. In this method, subdomain corners are not included in the approximation, thereby simplifying the subdomain connectivity. To allow for local refinement of the polynomial order or subdomain subdivision, non-conforming subdomains are treated by a mortar method. Spectral accuracy is shown in one- and two-dimensional linear and nonlinear problems. Application is made to four compressible flow problems: the Couette flow, a steady boundary layer over a flat plate, steady transonic flow in a nozzle, and unsteady flow over a cylinder at a Reynolds number of 75. 

## 1. INTRODUCTION

In a previous pair of papers, [17, 18], we introduced a staggered-grid spectral multidomain method for the Euler gas-dynamics equations. In that method, the solution unknowns were approximated in each subdomain at points wholly interior to the subdomain. The solution points were generated from the tensor product of Chebyshev–Gauss quadrature grids. The horizontal fluxes were collocated on the tensor product of the Gauss–Lobatto and Gauss grids, while the vertical fluxes were collocated on the tensor product of the Gauss and Gauss–Lobatto grids. The result was a fully staggered grid on which only the fluxes needed to be evaluated at subdomain interfaces or along boundaries. Coupling of subdomains was done by means of an approximate Riemann solver, which would choose between the two solution values available along an interface according to the normal inflow and outflow characteristics. On conforming grids, when the subdomains shared an entire side and the grid points matched up across an interface, the Riemann problems were solved directly [18]. When the grid was not conforming, due to a change of polynomial order across an interface,



0021-9991/98 $25.00

subdomain refinement, or both, a projection was made to a mortar on which the Riemann problem was solved [17].

For the Euler gas-dynamics equations, the staggered grid method is a simpler and more flexible multidomain method than one that enforces strong continuity on the solution at subdomain interfaces, e.g., [15]. It is simpler because subdomain corners are not included in the approximation, so special conditions do not need to be derived for them. Its flexibility comes from the fact that only the normal fluxes, not the flux derivatives or the tangential fluxes, need to be continuous across interfaces where subdomains meet. As a result, more general grids can be used. Finally, the method is conservative, preserves a constant free-stream solution, is temporally accurate to the order of the time differencing scheme, and retains spectral accuracy in space.

In this paper, we propose a staggered-grid spectral multidomain method for the solution of the compressible Navier–Stokes equations, based on the Euler method presented in [17, 18]. This time, the solution unknowns and gradients are approximated at the nodes of the Chebyshev–Gauss Quadrature rule, again, wholly within a subdomain. The total fluxes are evaluated on the tensor products of the Gauss–Lobatto and Gauss points. As before, to allow the flexibility to refine subdomains locally by increasing the polynomial order or by subdividing subdomains, a mortar method is used at subdomain interfaces [17].

Few spectral multidomain methods for the compressible Navier–Stokes equations have appeared in the literature to date. Hussaini and Zang [28] presented the first, solving a quasi-one-dimensional nozzle flow with two Chebyshev grids, one on each side of the throat. Later, Macaraeg and Streett [20] proposed a method that used continuity of the solution plus global flux conservation to impose the conditions necessary at subdomain interfaces. Hanley [10] followed with a method that used an alternating derivative technique to enforce strong solution and weak flux continuity. He showed that, for high accuracy, the method was significantly more efficient than a second order finite difference method.

Methods for multidimensional flows have been published only in the last few years. A method for two-dimensional flows that used a penalty method to enforce weak continuity of the viscous flux was presented by Kopriva [16]. This method, like those of [10, 20] enforced strong continuity of the solution and weak continuity of the flux.

Methods that do not require the continuity of the solution or the flux are the most recent [8, 11–13]. In [8], Giannakouros used a cell averaged Chebyshev method. Hesthaven presented a penalty method that applies a penalty for both the viscous and inviscid parts of the equations. The method was proven to be stable for the Legendre approximation. A triangular spectral element method was recently presented in [19].

We propose the staggered-grid method because it has advantages over methods based on a Lobatto grid, e.g., [10, 16], and extends easily the Euler methods of [17, 18]. It enforces weakly the continuity of the solution and the viscous fluxes at subdomain interfaces. Furthermore, staggering the solution unknowns and fluxes leads to a conservative algorithm. As in the inviscid case, the staggered-grid method imposes no smoothness requirements on the grid across subdomain interfaces, making possible the use of unstructured quadrilateral grids. More importantly, the staggering is done so that corners of subdomains are not included in the approximation. This makes special corner-point algorithms such as those described in [8, 11, 16] unnecessary. The result is a method for the Navier–Stokes equations that is easy to apply and has the same desirable properties as the Euler method.

The paper is organized as follows. We begin with the description of the approximation in one space dimension for a scalar equation in conservation form. The method is summarized

in Algorithm I. Two examples of the Burgers equation are solved to show spectral convergence of the solution. The two-dimensional algorithm is then described. Spectral accuracy is demonstrated for a linear scalar model problem and a linear system. As an example of the non-linear Navier–Stokes equations, we solve the Couette flow, and show that spectral accuracy is obtained. Finally, we apply the method to the solution of viscous flow over a flat plate in a subsonic free-stream, transonic flow in a two-dimensional nozzle, and subsonic flow over a circular cylinder. In the first problem, we compare the computed solution to spectral boundary-layer solutions. In the final two problems, comparisons to experimental data are made.

## 2. THE STAGGERED-GRID MULTIDOMAIN APPROXIMATION IN ONE SPACE DIMENSION

To motivate the approximation of the compressible Navier–Stokes equations, we consider the solution of scalar problems of the form

$$\begin{aligned} u_t + f_x(u) = 0, \qquad & x \in [a, b], t > 0 \\ u(x, 0) = u_0(x) & \\ u(a, t) = g^L(t), \qquad & u(b, t) = g^R(t), \end{aligned} \tag{1}$$

where the total flux consists of advective and diffusive parts, $f = f^a + f^v$.

For now, we assume that $\partial f^a/\partial u > 0$ and $f^v = -\nu u_x$. Alternatively, we consider application of Neumann conditions of the type $f_x(b, t) = \eta^R$, in place of a Dirichlet condition.

The domain $[a, b]$ is subdivided into multiple non-overlapping subdomains $\Omega^k = [a_k, b_k]$, $k = 1, 2, \ldots, K$, which are ordered left to right. A linear transformation $x(X) : \Omega^k \to [0, 1]$ is used to map the subinterval to the unit interval. Under the transformation, (1) can be written on each subinterval as

$$\tilde{u}_t + \tilde{f}_X(\tilde{u}) = 0, \tag{2}$$

where $\tilde{u} = x_X u$.

On each subdomain, we place two grids $X_j$ and $\bar{X}_{j+1/2}$, as described in [18]. These grids,

$$\begin{aligned} X_j &= \frac{1}{2}\left(1 - \cos\left(\frac{j\pi}{N}\right)\right), \qquad j = 0, 1, \ldots, N \\ \bar{X}_{j+\frac{1}{2}} &= \frac{1}{2}\left(1 - \cos\left(\frac{2j+1}{2N}\pi\right)\right), \qquad j = 0, 1, \ldots, N-1 \end{aligned} \tag{3}$$

correspond to the nodes of the Chebyshev Gauss–Lobatto and Gauss quadratures, respectively, mapped onto [0, 1]. We will call these the *Lobatto* and the *Gauss* grids for short. For the sake of discussion only, we assume the same number of points are used in each subdomain. Polynomial interpolants on these two grids can be represented in terms of the two sets of Lagrange interpolating polynomials

$$\begin{aligned} l_j &= \prod_{i=0, j\neq i}^{N} \left(\frac{\xi - X_i}{X_j - X_i}\right) \\ h_{j+\frac{1}{2}} &= \prod_{i=0, j\neq i}^{N-1} \left(\frac{\xi - \bar{X}_i}{X_{j+\frac{1}{2}} - X_{i+\frac{1}{2}}}\right). \end{aligned} \tag{4}$$

We denote the space of all polynomials of degree less than or equal to $N$ by $\mathsf{P}_N$. Then the $l_j$'s form a basis for $\mathsf{P}_N$, while the $h_j$'s form a basis for $\mathsf{P}_{N-1}$.

The motivation for the use of the two grids comes from the form of Eq. (1). If the flux and solution are represented by polynomials in $\mathsf{P}_N$ and $\mathsf{P}_{N-1}$, respectively, then the polynomial order of the time derivative of the solution matches that of the space derivative of the flux. A unique solution can then be obtained by collocating the equation at the $N$ Gauss points.

The solution is approximated on the Gauss grid. That is, we let $\bar{U}^k \in \mathsf{P}_{N-1}$ approximate $\tilde{u}$ on $\Omega^k$ where

$$\bar{U}^k(X) = \sum_{j=0}^{N-1} \bar{U}^k_{j+\frac{1}{2}} h_{j+\frac{1}{2}}(X) \in \mathsf{P}_{N-1}. \tag{5}$$

The flux, $\tilde{f}$, is approximated on the Lobatto grid by the polynomial

$$F^k(X) = \sum_{j=0}^{N} F^k_j l_j(X) \in \mathsf{P}_N. \tag{6}$$

Equations for the unknowns, $\bar{U}_{j+1/2}$, are derived by the usual collocation procedure [4]. Thus, we require that the residual vanish on the Gauss grid so that

$$\frac{\partial \bar{U}^k_{j+\frac{1}{2}}}{\partial t} + \frac{\partial F^k\left(\bar{X}_{j+\frac{1}{2}}\right)}{\partial X} = 0, \qquad j = 0, 1, \ldots, N-1. \tag{7}$$

The differentiation of the flux can be performed efficiently by matrix multiplication, see [18].

The flux, $F^k_j$, is computed by considering the advective and viscous fluxes separately. The procedure for the advective flux follows that of [18]. First, the solutions at the Lobatto points, $U^k_j$, are computed from the polynomial interpolant (5). Since the family of characteristics for the advective part of (1) runs from left to right, the values of $U^k_0$ are not used to compute the $F^k_0$. Instead, they are replaced by $g^L(t)$ for $k=0$, and by $U^{k-1}_N$ for $k \geq 1$, giving the approximation the proper domain of dependence.

The computation of the diffusive flux is done in two steps. When the solution is evaluated at the Lobatto points, the result can be written as a polynomial defined piecewise over the subdomains,

$$\bar{U}^k(X) = \sum_{j=0}^{N} \bar{U}^k(X_j) l_j(X). \tag{8}$$

It is unlikely that this polynomial will exactly satisfy the boundary conditions, or be continuous at the subdomain interfaces. To satisfy Dirichlet boundary conditions, the computed value at the boundary is replaced by the boundary condition. The lack of continuity at the interfaces poses a more difficult problem, because it is impossible to compute a unique first derivative at those points. To allow differentiation at interfaces, we construct a continuous piecewise polynomial from (8).

The computation of a continuous piecewise polynomial can be accomplished in several ways. One approach is to adjust the value of the solution at the interfaces so that the polynomial and its first derivative are continuous. Thus, one sets $U^k_j = \bar{U}^k(X_j)$ at the interior

points, treats the values at the subdomain interfaces as unknowns, and solves the tri-diagonal system of equations generated by

$$\begin{aligned} \sum_{j=0}^{N} d_{ij} U_j^{k-1} &= \sum_{j=0}^{N} d_{ij} U_j^k, \\ U_N^{k-1} &= U_0^k, \end{aligned} \qquad k = 1, 2, \ldots, N \tag{9}$$

for those unknown values. The advantage of this approach is that the resulting polynomial can be continuously differentiated across an interface. The disadvantage is that the procedure couples all subdomains together, not just nearest neighbors. In addition to complexity, particularly in multiple space dimensions, the coupling of all the subdomains affects the possible parallel efficiency of the algorithm.

An alternative approach is to seek a continuous piecewise polynomial, $U^k(X)$, that is closest, in some sense, to the $\bar{U}^k(X)$. If we require that the new continuous polynomial take on the values of $\bar{U}^k$ at the interior Lobatto points, $U^k$ can be written as

$$U^k = \bar{U}^k(X) + \alpha l_0(x) + \beta l_N(x). \tag{10}$$

It remains, then, to compute the coefficients $\alpha$ and $\beta$ so that $U^{k-1}(1) = U^k(0)$. In this paper, we choose $\alpha$ and $\beta$ so that they each have the smallest magnitude, i.e.,

$$\begin{aligned} \alpha &= \frac{1}{2}(\bar{U}^{k-1}(1) - \bar{U}^k(0)) \\ \beta &= \frac{1}{2}(\bar{U}^{k+1}(0) - \bar{U}^k(1)). \end{aligned} \tag{11}$$

This choice is equivalent to that made in [2, 8]. Using (10) and (11), only nearest neighbors are involved, and $U^k$ takes on the average value of the two $\bar{U}^k$ values at the interfaces. Note that the coefficients $\alpha$, $\beta$ are proportional to the jump in the solution across the interface. At physical boundaries where a Dirichlet condition is required, the boundary value replaces the value of the interpolant at the boundary point.

Once the continuous piecewise polynomial $U^k \in \mathsf{P}_N$ is found, the diffusive flux is computed. First, $U^k$ is differentiated subdomain by subdomain and the flux is evaluated at the Gauss points

$$\bar{F}^{v,k}_{j+\frac{1}{2}} = \nu \sum_{i=0}^{N} U_i^k l_i'\left(X_{j+\frac{1}{2}}\right), \tag{12}$$

which is natural, since the derivative is a polynomial of degree $N-1$. From this flux, it is necessary to reconstruct a continuous flux on the Lobatto grid. As above, we compute

$$F^k = \bar{F}^k(X) + \delta l_0(x) + \varepsilon l_N(x), \tag{13}$$

where

$$\begin{aligned} \delta &= \frac{1}{2}(\bar{F}^{k-1}(1) - \bar{F}^k(0)) \\ \varepsilon &= \frac{1}{2}(\bar{F}^{k+1}(0) - \bar{F}^k(1)). \end{aligned} \tag{14}$$

This time the coefficients are proportional to the jump in the diffusive flux across the interfaces. Thus, the procedure for finding a piecewise continuous flux is identical to that of computing the piecewise continuous solution. Neumann boundary conditions can be applied by setting the boundary values of the flux to the specified values (cf. [4]).

To summarize the staggered-grid procedure for the spatial approximation of Eq. (1), we have the following algorithm:

ALGORITHM I (Staggered-Grid, Scalar, 1D).

1. Re-construct $\bar{U}^k = [\bar{U}_{1/2}, \bar{U}_{3/2}, \ldots, \bar{U}_{N-\frac{1}{2}}]^k$ at the Lobatto points:

$$U^k = \mathbf{I}_N^k \bar{U}^k, \qquad \left(\mathbf{I}_N^k\right)_{i,j} = h_{j+\frac{1}{2}}^k(X_i)$$

2. Compute the advective flux:
   - Compute the flux values at internal points on the Lobatto grid

$$F_j^{a,k} = f^a\left(U_j^k\right), \qquad j = 1, \ldots, N$$

   - Apply boundary and interface conditions

$$F_0^{a,0} = f^a(g^L)$$
$$F_0^{a,k} = f^a\left(U_N^{k-1}\right)$$

3. Compute the Diffusive Flux
   - Average solutions at interfaces

$$U_0^k = U_N^{k-1} \leftarrow \frac{1}{2}\left(U_0^k + U_N^{k-1}\right)$$

   - Apply Dirichlet boundary conditions (As Necessary)

$$U_0^0 = g^L$$
$$U_N^K = g^R$$

   - Differentiate and compute flux at Gauss points

$$\bar{F}^{v,k} = \nu \mathbf{D} U^k, \qquad (\mathbf{D})_{i,j} = l_j'\left(\bar{X}_{i+\frac{1}{2}}\right)$$

   - Interpolate flux to Lobatto points

$$F^{v,k} = \mathbf{I}_N^k \bar{F}^{v,k}$$

   - Average flux values at interfaces

$$F_0^{v,k} = F_N^{v,k-1} \leftarrow \frac{1}{2}\left(F_0^{v,k} + F_N^{v,k-1}\right)$$

   - Apply Neumann boundary conditions (As Necessary)

$$F_0^{v,0} = \eta^L, \qquad F_N^{v,K} = \eta^R$$

4. Combine Fluxes to get the total flux

$$F_j^k = F_j^{a,k} + F_j^{v,k}, \qquad j = 0, 1, \ldots, N; k = 1, 2, \ldots, K$$

5. Differentiate Flux and evaluate on Gauss Grid

$$\frac{d}{dt}\bar{U}^k_{j+\frac{1}{2}} + (\mathbf{D}F^k)_{j+\frac{1}{2}} = 0, \qquad j = 0, 1, \ldots, N-1; k = 1, 2, \ldots, K$$

Algorithm I can be easily extended to the solution of systems of equations. The computation follows that of the scalar equation, except for the approximation of the advective flux at boundaries and subdomain interfaces, where it is necessary to allow waves to propagate in either direction. This is done by the procedure described in [18], where an approximate Riemann solver is used to resolve the jump in the solution at an interface.

Finally, the semi-discrete equation (7) is a system of ordinary differential equations for the solution at the Gauss points. To integrate in time, any standard time discretization procedure can be used. In order to minimize storage, we have chosen to use Runge–Kutta methods that require only $2N$ levels. In particular, we typically use either the third order Runge–Kutta method of Williamson [27] or the fourth order method of Carpenter and Kennedy [5].

### 2.1. Examples

As examples of the performance of the staggered grid approximation, we consider the solution of the Burgers equation with two initial and boundary conditions. The first problem, which has a travelling wave solution, was used as a test problem by Hesthaven [12] for his spectral penalty method. The second problem, which has a stationary near-shock solution, has been used as a test problem for a number of spectral multidomain methods [1, 12, 16].

#### 2.1.1. *Travelling Wave Problem*

As an example of the use of Algorithm I, we consider the solution of the boundary-value problem

$$\begin{aligned} \frac{\partial u}{\partial t} + \frac{\partial((1/2)u^2)}{\partial x} &= \nu \frac{\partial^2 u}{\partial x^2} \\ u(-\infty, t) &= b_{-\infty} \\ u(+\infty, t) &= b_{+\infty} \\ u(x, 0) &= -a \tanh\left(\frac{ax}{2\nu}\right) + c, \end{aligned} \tag{15}$$

where $c = (b_{-\infty} + b_{+\infty})/2$ and $a = (b_{-\infty} - b_{+\infty})/2$. The diffusion coefficient is chosen to be $\nu = 0.1$. We will present solutions on the interval $[-.5, 1.5]$ with four subdomains of equal size for times between zero and one. The time step was chosen so that the error is dominated by spatial rather than temporal error.

Figure 1 shows the computed propagating wave solution to (15) at four times, along with the exact solution. In each subdomain, eight Gauss points were used, for a polynomial approximation of degree seven. Figure 2 shows that the convergence of the unweighted $L^2$

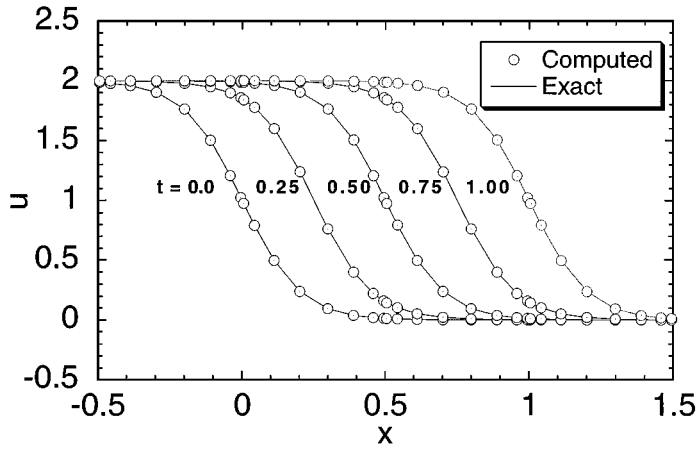


**FIG. 1.** Computed and exact travelling wave solutions to the Burgers equation.

error is exponential with polynomial order. To compute the unweighted error, the Clenshaw–Curtis rule [4] evaluated at twice the number of collocation points was used. For comparison, we have also plotted on Fig. 2 the errors obtained using the methods proposed in [12, 16]. The penalty method error shown here is for an unfiltered solution. Those presented in [12] were filtered with a cutoff frequency of $N/2$ to be able to take a larger time step, at the expense of convergence rate. The best accuracy and convergence rate is found for the method presented in [16]. However, unlike the staggered-grid and penalty methods, that method strongly enforces the continuity of the solution, while weakly enforcing the continuity of the flux. Both the penalty method of [12] and the staggered-grid method weakly enforce the continuity of the solution and the flux.

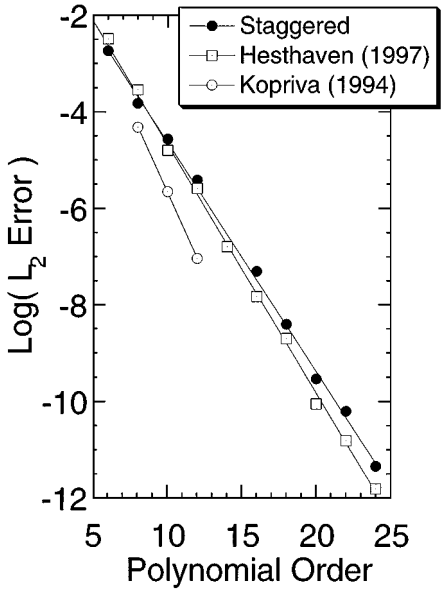


**FIG. 2.** Error convergence of three methods for the travelling wave solution to the Burgers equation.

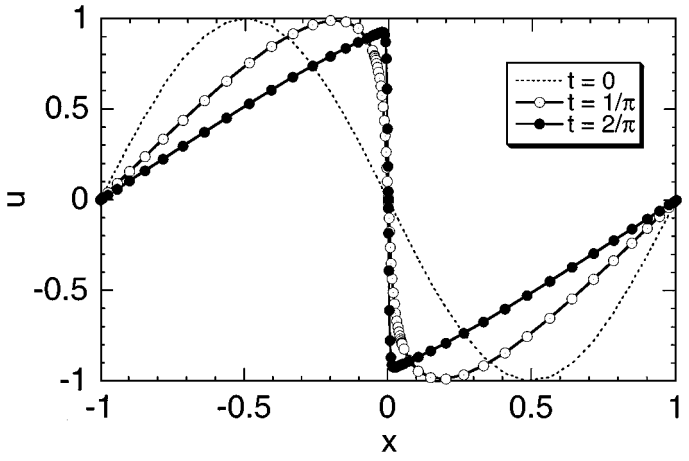


**FIG. 3.** Solution of the Burgers equation with a viscous shock.

### 2.1.2. *Viscous Shock Problem*

The next problem solves the Burgers equation on $[-1, 1]$, with $\nu = 1/100\pi$ and replaces the boundary and initial conditions in (15) by

$$\begin{aligned} u(-1, t) &= u(1, t) = 0 \\ u(x, 0) &= -\sin(\pi x). \end{aligned} \tag{16}$$

We solve the problem using four subdomains with interfaces at $x = -0.05$ and $x = 0.05$, the same subdomain decomposition considered in [1, 12, 16]. Again, the time step was adjusted so that the error was dominated by the spatial approximation.

Figure 3 shows the solution at three times, from the initial sine curve to the near discontinuity at $t = 2/\pi$. The first derivative of the solution is shown in Fig. 4. In both of these plots, the solution was interpolated to the Lobatto points using (10). A quantitative examination can be made using the analytic value of the maximum slope calculated in [1]

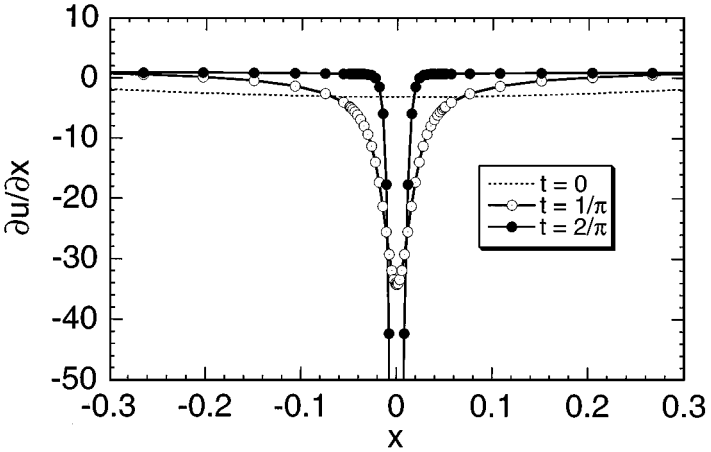


**FIG. 4.** Derivative of the viscous shock solution of the Burgers equation.

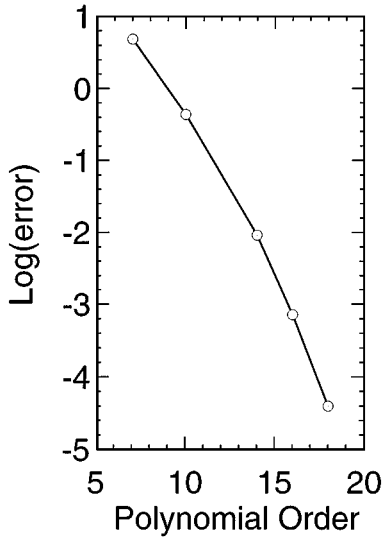


**FIG. 5.** Convergence of the derivative at $x = 0, t = 1.6037/\pi$ for the viscous shock solution of the Burgers equation.

to be 152.00516 at time $t = 1.6037/\pi$. Figure 5 shows the exponential convergence of the error in the derivative at $x = 0$ as a function of polynomial order, to the precision of the exact solution.

### 2.2. Properties of the Staggered-Grid Approximation

The procedure that computes the continuous solution and flux on the Lobatto grid from the Gauss grid values weakly enforces continuity of both the solution and the diffusive flux, which are the natural continuity conditions at the interfaces. To see this, consider the case of two subdomains, $L$ and $R$, $\nu$ constant, and $f^a(u) = u$. Assume that each subdomain has unit length. Then if (10) and (13) are substituted into (7), we find that the solution on the left subdomain satisfies the differential equation

$$\bar{U}^L_t + \bar{U}^L_X - \nu \bar{U}^L_{XX} = -\kappa l'_0 + \varepsilon l'_N + \nu(\alpha l''_0 + \beta l''_N), \tag{17}$$

where $\kappa = (g^L - \bar{U}^L(0))$ and $\alpha, \beta, \varepsilon$ are defined above. Thus, the boundary and continuity constraints are imposed on the left solution by the penalty term on the right of (17). A similar equation is satisfied $\bar{U}^R$ on the right.

In the single domain case, it is possible in the linear case to reconstruct exactly the standard Lobatto collocation solution from the staggered-grid solution. In the case of two Dirichlet boundary conditions, the staggered-grid solution satisfies exactly the PDE

$$\bar{U}_t + U_x - \nu U_{xx} = 0, \tag{18}$$

where $U = \bar{U} + \delta^L \ell_0 + \delta^R \ell_N$, $\delta^L = g^L - U(0, t)$, and $\delta^R = g^R - U(1, t)$. If we replace $\bar{U}$ in (18) we get

$$U_t + U_x - \nu U_{xx} = -\delta^L_t \ell_0 - \delta^R_t \ell_N. \tag{19}$$

Finally, the right hand side of (19) vanishes at the interior Lobatto points, so that $U$ as defined above satisfies the standard Lobatto collocation approximation

$$U_t|_j + U_x|_j - \nu U_{xx}|_j = 0, \qquad j = 1, 2, \ldots, N-1$$
$$U_0 = g^L, \qquad U_N = g^R.$$

Thus, we can re-construct the standard Lobatto collocation solution, $U$, from the staggered-grid solution, $\bar{U}$, by the formula

$$U = \bar{U} + (g^L - U(0,t))\ell_0 + (g^R - U(1,t))\ell_N.$$

A study of the eigenvalues of the global second-order differentiation operator was made for two subdomains and $4 \le N \le 40$. For equally sized subdomains with the same number of points in each subdomain, the eigenvalues are real and non-positive for both Dirichlet–Dirichlet and Dirichlet–Neumann boundary conditions. Some of the eigenvalues become complex, but remain non-positive, if the number of points or the subdomain sizes differ.

The eigenvalue with the largest magnitude is real in all cases tested. For two equally sized subdomains with the same number of points in each, that eigenvalue grows as $0.058N^4$ for the range considered, compared to the asymptotic value of $0.047N^4$ for standard Chebyshev collocation [4]. Thus, the time step restriction for the staggered-grid method is comparable to that of the standard Chebyshev collocation. It can be as much as an order of magnitude larger than that required by the method presented in [16], and is about 60% of the time step reported for the penalty method [13].

In the Dirichlet–Dirichlet case, the diffusion approximation has a computational mode corresponding to the highest frequency that can be supported on the Gauss grid. This mode is associated with a zero eigenvalue of the global second-order diffusion operator. It can be seen easily, for example, if we take $\bar{U} \in \mathsf{P}_0$, i.e., constant in each subdomain. If we replace $j + \frac{1}{2}$ by $j$, then the staggered grid approximation for the advection-diffusion equation on a uniform grid reduces to a recognizable finite difference approximation:

$$\frac{d}{dt}\bar{U}_j + \frac{\bar{U}_j - \bar{U}_{j-1}}{\Delta x} - \nu \frac{\bar{U}_{j+2} - 2\bar{U}_j + \bar{U}_{j-2}}{4\Delta x^2} = 0.$$

Thus it is upwind in the advective terms and centered in the diffusion terms. In the absence of advection, the method does not damp the $2\Delta x$ mode. There is no such computational mode in the Dirichlet–Neumann case, which corresponds to the inflow/outflow situation. Since the problems we have been concerned with are advection dominated, we have not observed any need to filter the highest frequency mode in any of the problems reported in this paper.

Finally, a desirable feature of the staggered grid algorithm is that it is globally conservative, as in the purely advective case [18]. To show this, we define the quadrature

$$\int_0^1 v(x)\,dx = \sum_{j=0}^{N-1} v_{j+\frac{1}{2}} w_{j+\frac{1}{2}} \qquad \forall v \in \mathsf{P}_{N-1}$$
$$w_{j+\frac{1}{2}} = \int_0^1 h_{j+\frac{1}{2}}(X)\,dX. \tag{20}$$

For each $j$, we multiply (7) by $w_{j+1/2}$ and sum over all points and all subdomains to get

$$\sum_{k=1}^{K}\sum_{j=0}^{N-1} w_{j+\frac{1}{2}}\left(\frac{\partial \bar{U}^k_{j+\frac{1}{2}}}{\partial t}+\frac{\partial F^k\left(\bar{X}_{j+\frac{1}{2}}\right)}{\partial X}\right). \tag{21}$$

Since both $\bar{U}(X)$ and $\partial F/\partial X \in \mathsf{P}_{N-1}$, the quadrature is exact. If we now evaluate the integrals and use the fact that the flux is continuous at the subdomain interfaces, we get

$$\frac{d}{dt}\left\{\sum_{k=1}^{K}\int_{\Omega^k}\bar{U}^k x^k_X\,dX\right\} = F^1(0)-F^k(1). \tag{22}$$

## 3. THE STAGGERED-GRID APPROXIMATION IN TWO SPACE VARIABLES

### 3.1. The Navier–Stokes Equations

In two space dimensions, we write the Navier–Stokes equations as

$$\mathbf{Q}_t+\mathbf{F}^a_x+\mathbf{G}^a_y=\frac{1}{\mathrm{Re}}\left(\mathbf{F}^v_x+\mathbf{G}^v_y\right), \tag{23}$$

where $Q$ is the vector of solution unknowns and $F(Q)$ and $G(Q)$ are the advective flux vectors

$$\mathbf{Q}=\begin{bmatrix}\rho\\ \rho u\\ \rho v\\ \rho e\end{bmatrix},\qquad \mathbf{F}^a=\begin{bmatrix}\rho u\\ p+\rho u^2\\ \rho uv\\ u(\rho e+p)\end{bmatrix},\qquad \mathbf{G}^a=\begin{bmatrix}\rho v\\ \rho uv\\ p+\rho v^2\\ v(\rho e+p)\end{bmatrix}. \tag{24}$$

The pressure, $p$, is defined through the ideal gas relation $\rho e = p/(\gamma-1)+\rho(u^2+v^2)/2$, where $\gamma = 7/5$. Finally, we write the viscous fluxes as

$$\mathbf{F}^v=\begin{bmatrix}0\\ \tau_{xx}\\ \tau_{yx}\\ u\tau_{xx}+v\tau_{yx}+\frac{\gamma\kappa}{(\gamma-1)\mathrm{Pr}}T_x\end{bmatrix},\qquad \mathbf{G}^v=\begin{bmatrix}0\\ \tau_{xy}\\ \tau_{yy}\\ u\tau_{xy}+v\tau_{yy}+\frac{\gamma\kappa}{(\gamma-1)\mathrm{Pr}}T_y\end{bmatrix}, \tag{25}$$

where, under the Stokes hypothesis,

$$\begin{aligned}\tau_{xx}&=2\mu(u_x-(u_x+v_y)/3)\\ \tau_{yy}&=2\mu(v_y-(u_x+v_y)/3)\\ \tau_{yx}&=\tau_{xy}=\mu(v_x+u_y)\end{aligned} \tag{26}$$

Equations (23)–(26), have been scaled with respect to reference quantities $L$, for length, $p_0$ and $\rho_0$ for pressure and density, and $\mu_0$ for viscosity. Velocities are scaled by $\sqrt{p_0/\rho_0}$, proportional to the reference sound speed, rather than a reference velocity. Lengths are non-dimensionalized by a length scale $L$, and time by $L/\sqrt{p_0/\rho_0}$. Under these scalings, the equation of state becomes $p=\rho T$. The non-dimensional parameters are the Reynolds number, $\mathrm{Re}=Lp_0/(\mu_0\sqrt{p_0/\rho_0})$ and the Prandtl number $\mathrm{Pr}=\kappa_0/(R\mu_0)$. We will use $\mathrm{Pr}=0.72$ throughout this paper. The Reynolds number differs from the usual definition because the equations are scaled to a reference sound speed, rather than the more common reference velocity. In this paper, we will refer to the more common Reynolds number based on a

reference velocity by $\mathrm{Re}_v = \sqrt{\gamma} M_0 \mathrm{Re}$, where $M_0$ is the reference Mach number. Finally, unless otherwise noted, we compute the viscosity coefficient, $\mu$, by the Sutherland law [25].

### 3.2. Domain Decomposition

The domain decomposition used in this paper is described in detail in Refs. [17, 18]. In summary, the two-dimensional domain is subdivided into $K$ quadrilateral subdomains, $\Omega^k$. We assume that these subdomains do not move in time. Subdomains can be conforming or non-conforming. A subdomain is conforming if it intersects its neighbors either along an entire face, or at a corner. Non-conforming subdomain decompositions that we consider here occur when subdomains in an originally conforming subdomain decomposition have been subdivided locally (cf. [17, 21]). The subdomains are mapped onto the unit square $(X, Y) \in [0, 1] \times [0, 1]$ by an isoparametric mapping. Under the mapping used in each subdomain, the Navier–Stokes equations become

$$\frac{\partial \tilde{\mathbf{Q}}}{\partial t} + \frac{\partial \tilde{\mathbf{F}}}{\partial X} + \frac{\partial \tilde{\mathbf{G}}}{\partial Y} = 0, \tag{27}$$

where

$$\begin{aligned} \tilde{\mathbf{Q}} &= J\mathbf{Q} \\ \tilde{\mathbf{F}} &= y_Y\left(\mathbf{F} - \frac{1}{\mathrm{Re}}\mathbf{F}^v\right) - x_Y\left(\mathbf{G} - \frac{1}{\mathrm{Re}}\mathbf{G}^v\right) \\ \tilde{\mathbf{G}} &= -y_X\left(\mathbf{F} - \frac{1}{\mathrm{Re}}\mathbf{F}^v\right) + x_X\left(\mathbf{G} - \frac{1}{\mathrm{Re}}\mathbf{G}^v\right) \\ J &= x_X y_Y - x_Y y_X. \end{aligned} \tag{28}$$

### 3.3. The Staggered Grid

The same fully staggered grid is used for the Navier–Stokes equations in two space dimensions as was used for the Euler equations [18]. A diagram of the staggered grid on the image of a subdomain is reproduced in Fig. 6. Points of type a represent the Gauss/Gauss points, $(\overline{X}_{i+1/2}, \overline{Y}_{j+1/2})$ $i = 0, 1, \ldots, N-1$; $j = 0, 1, \ldots, M-1$. Points of type b represent the

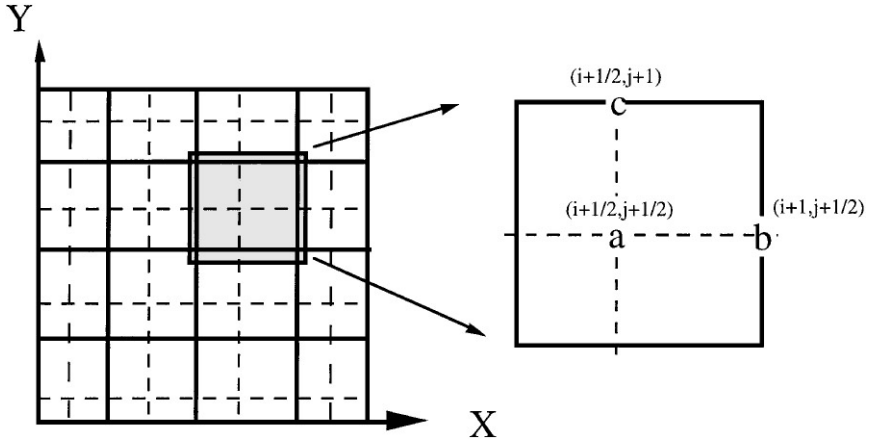


**FIG. 6.** Diagram of the fully staggered grid used in two space dimensions.

**TABLE 1**
**Quantities and Their Collocation Points on the Staggered Grid Shown in Fig. 6**

| Point | Quantity | Polynomial Order |
|---|---|---|
| a | $\tilde{\mathbf{Q}}$ $\nabla u$ $\nabla v$ $J$ $\nabla T$ | $\mathsf{P}_{N-1,M-1}$ |
| b | $\tilde{\mathbf{F}}$ $y_Y$ $x_Y$ | $\mathsf{P}_{N,M-1}$ |
| c | $\tilde{\mathbf{G}}$ $y_X$ $x_X$ | $\mathsf{P}_{N-1,M}$ |

Lobatto/Gauss points, $(X_i, Y_{j+1/2})$ $i = 0, 1, \ldots, N$; $j = 0, 1, \ldots, M-1$, while those of type c represent the Gauss/Lobatto points, $(X_{i+1/2}, Y_j)$ $i = 0, 1, \ldots, N-1$; $j = 0, 1, \ldots, M$. For simplicity, we will often refer to the Gauss/Gauss points as the "cell centers," and to points b and c as "cell faces." It is to be understood that the Gauss points do not actually fall halfway between the Lobatto points and that the "cells" within the subdomains are not independent of each other, as in a finite volume method. Polynomials that interpolate the Gauss/Gauss points are polynomials in $\mathsf{P}_{N-1,M-1} \equiv \mathsf{P}_{N-1} \times \mathsf{P}_{M-1}$, while polynomials at the Gauss/Lobatto and Lobatto/Gauss points are in $\mathsf{P}_{N-1,M}$ and $\mathsf{P}_{N,M-1}$, respectively.

The motivation for the staggered grid shown is that polynomial approximations of degree $N$ that are to be differentiated once are represented on a Lobatto grid. Other quantities are of degree $N-1$, and are represented on a Gauss grid. Thus, solution unknowns are collocated at points of type a, while fluxes are collocated at points of type b and c. The horizontal flux, $\tilde{F}$, for instance, is differentiated with respect to $X$ in (27), and hence is approximated by a polynomial in $\mathsf{P}_{N,M-1}$ that interpolates values on the Lobatto/Gauss grid. A list of the quantities and where they are collocated is shown in Table 1.

The solution is collocated at the Gauss/Gauss points, and is denoted by $\tilde{Q}_{i+1/2,j+1/2}$, $i = 0, 1, \ldots, N-1$; $j = 0, 1, \ldots, M-1$. To compute the fluxes, we write the polynomial

$$Q(X,Y) = \sum_{i=0}^{N-1} \sum_{j=0}^{M-1} \frac{\tilde{Q}_{i+\frac{1}{2},j+\frac{1}{2}}}{J_{i+\frac{1}{2},j+\frac{1}{2}}} h_{i+\frac{1}{2}}(X) h_{j+\frac{1}{2}}(Y). \tag{29}$$

From this polynomial, we compute the solution values at the "cell faces," $Q_{i+1,j+1/2} = Q(X_{i+1}, Y_{j+1/2})$ and $Q_{i+1/2,j+1} = Q(X_{i+1/2}, Y_{j+1})$. Though the solutions at the cell faces still describe a polynomial in $\mathsf{P}_{N-1,M-1}$, application of boundary and interface conditions add extra degrees of freedom to increase the polynomial order.

The advective fluxes are computed from the cell face values exactly as described in [18]. It remains, then, to describe how the viscous fluxes, (25), are approximated.

To compute the viscous fluxes, we first compute cell centered values of the gradients from the cell face values. This is done in conservative form to be consistent with the computation of the inviscid flux. For instance,

$$\begin{aligned} u_x|_{i+\frac{1}{2},j+\frac{1}{2}} &= \frac{1}{J_{i+\frac{1}{2},j+\frac{1}{2}}} [(y_Y u)_X - (y_X u)_Y]_{i+\frac{1}{2},j+\frac{1}{2}} \\ u_y|_{i+\frac{1}{2},j+\frac{1}{2}} &= \frac{1}{J_{i+\frac{1}{2},j+\frac{1}{2}}} [-(x_Y u)_X + (x_X u)_Y]_{i+\frac{1}{2},j+\frac{1}{2}}. \end{aligned} \tag{30}$$

Derivatives of the vertical velocity and the temperature are computed in the same way. We note that evaluating the velocity derivatives at the cell centers has two desired effects.

First, it makes the evaluation of the divergence consistent with that used in the continuity equation. Second, the evaluation of the viscous fluxes will not require the use of subdomain corner points.

Once the derivative quantities are evaluated at the cell centers, we can define a polynomial interpolant of the form (29), so these quantities can be evaluated at the cell faces. From the cell face values, the viscous fluxes are computed and combined with the inviscid flux to get the total flux.

In summary, the procedure for computing the viscous fluxes is as follows: The solution, defined at the cell centers, is interpolated to the faces by (29). Dirichlet conditions are applied at subdomain boundaries (see below). Cell centered quantities of $\nabla u$, $\nabla v$, $\nabla T$ are computed from the face values using (30). The gradients are then interpolated back onto the faces and Neumann conditions are applied at subdomain boundaries (see below). Finally, the viscous flux is computed from values defined on cell faces. This entire procedure can be efficiently implemented as a series of matrix-vector operations.

### 3.4. Interface and Boundary Conditions

To describe how we compute the interface and boundary conditions using the staggered grid, we will refer to Fig. 7. The solution values, velocity, and temperature gradients are collocated interior to a subdomain at the cell centers, marked by solid circles. Along subdomain interfaces, one of the two flux components must be evaluated using either a boundary condition or an interface condition. Note that there is no connectivity between subdomains 1 and 4 or 2 and 3. The fact that corner conditions are not required by the approximation leads to a significant simplification in the specification of boundary and interface conditions [18].

#### 3.4.1. *Boundary Conditions*

Boundary conditions are applied at cell faces. Referring to Fig. 7, we see that only one flux component is needed on any subdomain side. Thus, the implementation of a

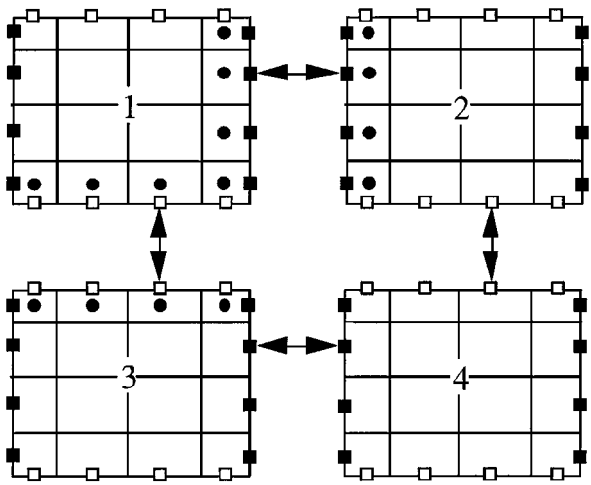


**FIG. 7.** Diagram of the connectivity between four subdomains showing points where the solutions are collocated (solid circles), horizontal fluxes (solid box), and vertical fluxes (open box).

boundary condition follows that of a one-dimensional approximation. It is sufficient for the discussion that follows, then, to assume a square subdomain where the horizontal flux $\mathbf{F} = \tilde{\mathbf{F}}$, and to describe the computation of just the horizontal flux. In this section, we will discuss the implementation of inflow, outflow, adiabatic wall, and isothermal wall boundary conditions.

*Inflow/outflow boundaries.* Since the advective and viscous terms are treated separately in the staggered-grid method, inflow and outflow boundary conditions are easy to apply. The separate treatment of the fluxes has the additional benefit that the Euler inflow and outflow boundary conditions are recovered as the Reynolds number tends to infinity. Computation of the inviscid fluxes follows that of the Euler method [18], where an external state (usually a mean flow) is specified, and the Riemann solver is used to sort out the characteristic relations. The additional boundary conditions required by the Navier–Stokes equations are applied to the computation of the viscous flux as Neumann conditions, as described in Algorithm I.

Well-posed inflow-outflow boundary conditions for the Navier–Stokes equations that reduce to well-posed conditions in the inviscid limit have been derived in a number of papers [7, 9, 13, 22]. See also [23]. In the calculations shown here, we have used the non-linear conditions proposed in [7].

The appropriate viscous flux conditions depend on whether the flow is supersonic or subsonic. At supersonic outflow points, we specify $T_x$, $\tau_{xx}$, and $\tau_{yx}$ as Neumann conditions. At supersonic inflow points, no viscous flux conditions are applied. Rather, the inflow values of the solution are imposed as Dirichlet conditions on the inflow faces. At subsonic outflow points, we specify $T_x$ and $\tau_{yx}$, again, as Neumann conditions. Finally, at subsonic inflow points, we specify $\tau_{xx}$. If the boundary values of these quantities are known, as they are, for instance, in the Couette flow test problem discussed below, the exact value is used. If they are not known, as is generally the case, an approximation must be made. In our calculations, we set the required values to zero, which gives an error proportional to 1/Re.

*Wall boundaries.* At solid wall boundaries, we impose no-slip conditions for the velocity and either an adiabatic or isothermal condition on the temperature. As before, the computations of the viscous and inviscid fluxes are treated separately. The no-slip condition is enforced on the inviscid flux by imposing a flow on the other side of the wall with equal but opposite velocity. If the wall is adiabatic, the total energy is the same on either side of the wall. If the wall is isothermal, then the internal energy is set using the wall boundary temperature. Thus, if the evaluation of (29) at a vertical wall point $(X_0, Y_{j+\frac{1}{2}})$ gives $Q_{0,j+1/2} = [\rho, \rho u, \rho v, \rho e]^T_{0,j+1/2}$, we define the external state $Q_e = [\rho, -\rho u, -\rho v, \rho e]^T$ for an adiabatic wall. For an isothermal wall, we replace $\rho e_{0,j+1/2}$ by $\rho e = \rho T_{wall}/(\gamma - 1) + ((\rho u)^2 + (\rho v)^2)/2\rho$. The advective flux is then simply

$$F^a_{0,j+\frac{1}{2}} = \mathcal{F}\left(Q_e, Q_{0,j+\frac{1}{2}}\right) \tag{31}$$

The viscous flux at a wall is computed in two stages, as described above (cf. Algorithm I). After the solution is re-constructed at the cell faces, the Dirichlet conditions $\rho u_{0,j+1/2} = \rho v_{0,j+1/2} = 0$ are set. If the wall is isothermal, then $T_{0,j+1/2} = T_{wall}$ is set. After the quantities $\nabla u$, $\nabla v$, $\nabla T$ are interpolated back onto the faces, Neumann conditions are applied as necessary. In particular, if the wall is adiabatic, we set $(T_x)_{0,j+1/2} = 0$. Once the viscous flux is evaluated it is added to the advective flux to obtain the total flux.

### 3.4.2. *Interface Conditions*

Interface conditions are also applied at cell faces. Referring to Fig. 7, we see again that only one flux component is needed on any subdomain side. Thus, as in the computation of the inviscid flux, the implementation of an interface condition follows that of a one-dimensional approximation. We consider in this section, then, the vertical interface between two square subdomains, such as that between subdomains one and two in Fig. 7 so that horizontal flux $\mathbf{F} = \tilde{\mathbf{F}}$, and describe the computation of just the horizontal flux.

*Conforming interfaces.* When two subdomains share an entire side and the collocation points match, as depicted in Fig. 7, we say that the interface between the two is conforming. In such cases, it is straightforward to enforce the interface conditions. The re-construction from the cell centers to the faces gives two solution values at each point along an interface, one from each contributing subdomain. To compute the inviscid flux at such points, an approximate Riemann solver is used to resolve waves propagating across the interface [18].

The procedure for computing the normal viscous flux at an interface is an extension of the one-dimensional Algorithm I. A continuous polynomial approximation is first computed by averaging the two solution values at each interface point. Then the solution gradients are interpolated onto the cell faces. Finally, the quantities $\tau_{xx}$, $\tau_{yx}$, and $T_x$ averaged before computing the viscous fluxes.

*Non-conforming interfaces.* If local refinement of the grid is desired, either by increasing the polynomial order within a single subdomain, or by subdividing a subdomain into multiple ones, it is not possible to perform the pointwise operations described above to compute the advective and viscous fluxes along the interfaces. Nevertheless, the ability to subdivide or refine subdomains locally can significantly decrease the computational cost for a desired accuracy [17, 21].

When the grid is refined locally, the interfaces between the subdomains become non-conforming. We will consider two types of non-conforming interfaces in this paper. The first, which we will call *order refinement*, occurs when the subdomains themselves are conforming, but the approximations along a face differ in polynomial order. The second, which we will call *subdivision*, occurs when one of two conforming subdomains is subdivided into multiple subdomains (see Fig. 8). Of course, it is possible for an interface to have both subdivision and order refinement.

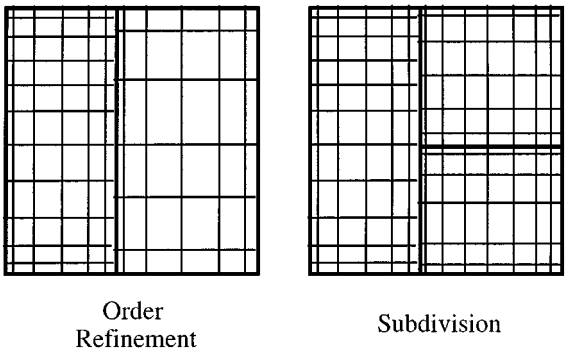


**FIG. 8.** Diagram of order refinement and subdivision type non-conforming interfaces.

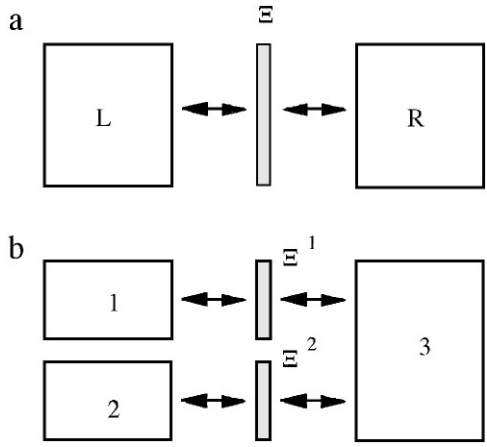


**FIG. 9.** Mortar topologies for order refinement (a) and subdivision (b).

We treat non-conforming interfaces by a mortar method. In a mortar method, a one-dimensional construct called a mortar is used to connect neighboring subdomains. The method was introduced for the incompressible Navier–Stokes equations in the late 1980s [21]. It was used for non-conforming approximations for the Euler gas-dynamics equations in [17].

The mortar approximation that we will use here was described in [17]. In this method, communication between subdomains occurs only through the intermediary mortars. When a subdomain interface is non-conforming, a projection of the solution (or viscous stress tensor) is made onto the mortars. A unique quantity is then computed on the mortars. Once the quantities are computed on the mortars, they are projected back onto the subdomain faces. Figure 9 shows the mortar topologies considered.

The projection operators that take values defined on subdomain faces onto the mortars and back are derived in [17]. The forward projection represents the integral least squares approximation of the solution from the face to the mortar. The reverse projection represents the least squares approximation of the solution from the mortar onto the face. The polynomial orders along the mortars are chosen to be of sufficient degree so that the projection of the solution onto the mortar and back returns the original solution. See [17] for details.

The inviscid flux is computed exactly as described in [17]. Once the projection onto a mortar is made from each subdomain, the Riemann problem is solved pointwise, as if the mortar were a conforming interface. The resulting flux is then projected back onto the subdomain faces.

The computation of the viscous flux is treated in a similar fashion. The solutions projected onto the mortars are averaged and then projected back onto the subdomain faces. From that point, the method proceeds as in the conforming case, until it is time to compute the viscous flux at the interface. At that time, the viscous stress tensors are projected onto the mortars, averaged pointwise as if the interface were conforming, and then projected back onto the faces.

### 3.5. Discretization of the Equations

Once the total fluxes are evaluated on the Gauss/Lobatto and Lobatto/Gauss grids, the spatial discretization is computed. From the grid point values of the fluxes, the interpolating

polynomials

$$\begin{aligned}\tilde{F}(X,Y) &= \sum_{i=0}^{N}\sum_{j=0}^{M-1}\tilde{F}_{i,j+\frac{1}{2}}l_i(X)h_{j+\frac{1}{2}}(Y) \in \mathsf{P}_{N,M-1}\\ \tilde{G}(X,Y) &= \sum_{i=0}^{N-1}\sum_{j=0}^{M}\tilde{G}_{i+\frac{1}{2},j}h_{i+\frac{1}{2}}(X)l_j(Y) \in \mathsf{P}_{N-1}\end{aligned} \tag{32}$$

are formally constructed. These polynomials are differentiated and evaluated at the Gauss/Gauss points to give pointwise values of the derivatives

$$\begin{aligned}\left.\frac{\partial\tilde{F}}{\partial X}\right|_{i+\frac{1}{2},j+\frac{1}{2}} &= \sum_{i=0}^{N-1}\tilde{F}_{i,j+\frac{1}{2}}l_i'\left(\bar{X}_{i+\frac{1}{2}}\right)\\ \left.\frac{\partial\tilde{G}}{\partial Y}\right|_{i+\frac{1}{2},j+\frac{1}{2}} &= \sum_{j=0}^{M-1}\tilde{F}_{i+\frac{1}{2},j}l_j'\left(\bar{Y}_{y+\frac{1}{2}}\right).\end{aligned} \tag{33}$$

Then the semi-discrete equation for the solution unknowns is

$$\left.\frac{d\tilde{Q}}{dt}\right|_{i+\frac{1}{2},j+\frac{1}{2}} + \left[\frac{\partial\tilde{F}}{\partial X}+\frac{\partial\tilde{G}}{\partial Y}\right]_{i+\frac{1}{2},j+\frac{1}{2}} = 0, \qquad \begin{cases} i = 0,1,\ldots,N-1\\ j = 0,1,\ldots,M-1.\end{cases} \tag{34}$$

Finally, Eq. (34) is integrated by a two-level Runge–Kutta scheme.

The advantage of writing the Navier–Stokes equations in the discrete form (34) is that the approximation satisfies the same properties as the approximation for the Euler equations. First, the method is conservative, in the sense that the rate of change of the integral of $\tilde{Q}$ over the entire domain depends only on the external boundary fluxes. This holds true for both conforming and non-conforming interfaces. See [17, 18] for the derivations. The method also remains free-stream preserving, meaning that if the solution is constant, it remains constant for all time. The arguments presented in [17, 18] still hold since free-stream preservation depends only on the approximation of the geometry, which is identical here.

In summary, we have the following algorithm for the solution of the compressible Navier–Stokes equations:

ALGORITHM II (Navier–Stokes Equations, 2D, non-conforming).

1. Re-construct Gauss/Gauss point values to the Lobatto/Gauss and Gauss/Lobatto points.
2. Compute the Advective Flux:
   - Compute the flux values at internal points on the Lobatto grid
   - Apply boundary conditions
   - Compute interface fluxes
     (a) Project interface solution values onto mortars
     (b) Compute the mortar fluxes
     (c) Project mortar fluxes back onto subdomain faces
3. Compute the Diffusive Flux
   - Compute interface solution values
     (a) Project interface solution values onto mortars
     (b) Average mortar values
     (c) Project solution averages back onto subdomain faces

- Apply Dirichlet boundary conditions to solution (As Necessary)
- Differentiate and compute gradients at Gauss points
- Interpolate flux to Lobatto points
- Compute interface viscous fluxes
  (a) Project interface flux values onto mortars
  (b) Average mortar values of viscous flux
  (c) Project flux averages back onto subdomain faces
- Apply Neumann boundary conditions (As Necessary)

4. Combine fluxes to get the total flux
5. Differentiate flux and integrate solution on Gauss grid

## 4. EXAMPLES

In this section, we use the method to compute two-dimensional solutions of linear model problems and the Navier–Stokes equations. Spectral accuracy is demonstrated for a linear scalar advection-diffusion problem and for a linear system. As an example of the non-linear Navier–Stokes equations for which the exact solution is known, we solve the Couette flow, and show that spectral accuracy is obtained. We then apply the method to the solution of viscous flow over a flat plate in a subsonic free-stream, transonic flow in a two-dimensional nozzle, and subsonic flow over a circular cylinder. In the first problem, we compare the computed solution to a spectral boundary-layer solution. In the final two problems, comparisons to experimental data are made.

### 4.1. Linear Model Problems

4.1.1. *Conforming Grid Solutions*

*A scalar model problem.* To explore the performance of the method, we first solve the scalar advection-diffusion equation

$$u_t + 0.2u_x + 0.5u_y = 0.1(u_{xx} + u_{yy}) + S \tag{35}$$

on the square $\Omega = [0, 2] \times [0, 2]$. The source term and boundary conditions are chosen so that the exact steady solution is

$$u = \frac{1}{3}\sum_{j=1}^{3} e^{-\gamma_j((x-x_j)^2+(y-y_j)^2)}, \tag{36}$$

where $\gamma_1 = 5$, $\gamma_2 = \gamma_3 = 10$, and $(x_1, y_1) = (1.0, 0.75)$, $(x_2, y_2) = (0.5, 1.5)$, $(x_3, y_3) = (1.5, 1.5)$. This problem is identical to that considered by Hesthaven [11], so that direct comparisons of the two methods can be made. We solved the problem on the 16 equal sized subdomains shown in Fig. 10, which Hesthaven concluded was the optimal number of subdomains for efficiency on this problem. Shown also in Fig. 10 is a steady solution calculated on that grid.

Convergence of the solution with polynomial order, $N$, is shown in Fig. 11. Shown are the errors for the staggered grid method, the penalty method, and the Lobatto grid method of [16]. All three are exponentially convergent.

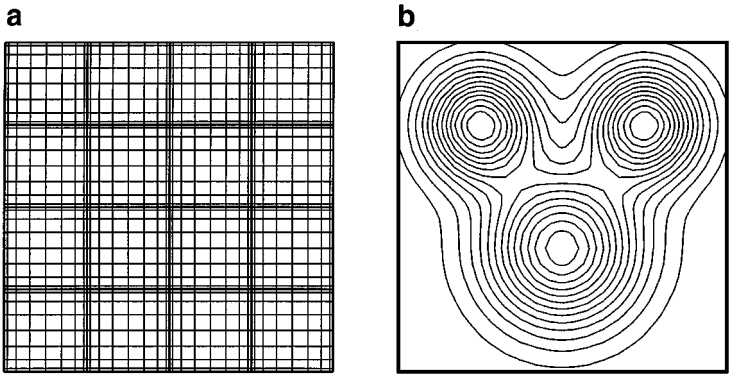


**FIG. 10.** Grid (a) and steady solution (b) for the linear advection-diffusion model problem.

*A linear system.* As a model that more closely represents the Navier–Stokes equations, we consider also the solution of the parabolic system of equations

$$\frac{\partial \mathbf{Q}}{\partial t} + \frac{\partial \mathbf{F}}{\partial x} + \frac{\partial \mathbf{G}}{\partial y} = \mathbf{S}(x, y), \tag{37}$$

where $\mathbf{F} = \mathbf{F}^a - \mathbf{F}^v$, $\mathbf{G} = \mathbf{G}^a - \mathbf{G}^v$, and $\mathbf{S}$ is a source term. In Eq. (37) we use

$$\begin{gathered}\mathbf{Q} = \begin{bmatrix} u \\ v \end{bmatrix}, \qquad \mathbf{F}^a = \begin{bmatrix} .3 & .6 \\ .6 & .3 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}, \qquad \mathbf{G}^a = \begin{bmatrix} .6 & .8 \\ .8 & .6 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} \\ \mathbf{F}^v = \begin{bmatrix} \frac{\nu}{3}\left(4\frac{\partial u}{\partial x} - 2\frac{\partial v}{\partial y}\right) \\ \nu\left(\frac{\partial v}{\partial x} + \frac{\partial u}{\partial y}\right) \end{bmatrix}, \qquad \mathbf{G}^v = \begin{bmatrix} \nu\left(\frac{\partial v}{\partial x} + \frac{\partial u}{\partial y}\right) \\ \frac{\nu}{3}\left(4\frac{\partial v}{\partial y} - 2\frac{\partial \mu}{\partial x}\right) \end{bmatrix}.\end{gathered} \tag{38}$$

We take $\nu = 0.1$. System (37) is designed to test the ability of the algorithm to handle problems with cross derivatives. The source term, $S$, was chosen so that the exact, steady, solutions for $u$ and $v$ were

$$\begin{aligned} u &= e^{\sin(\pi x)+\cos(\pi y)} \\ v &= \frac{1}{2}\left(e^{\alpha[(x-x_1)^2+(y-y_1)^2]} + e^{\alpha[(x-x_2)^2+(y-y_2)^2]}\right), \end{aligned}$$

where $\alpha$ was chosen to be $-10$, $(x_1, y_1) = (0.8, 0.6)$, and $(x_1, y_1) = (0.3, 0.4)$.

The system was computed on the grid shown in Fig. 12, which is bounded by $[0, 0.4] \times [-0.6, 0.6]$, and is composed of 18 subdomains. This grid was chosen to show that the method can solve on complex, multiply connected geometries without regard to the number of subdomains that come together at a point. The computed solutions for $u$ and $v$ are shown in Fig. 13. Convergence of the error, shown in Fig. 14, is exponential.

### 4.1.2. *Non-conforming Approximations*

Non-conforming approximations also exhibit exponential convergence as the polynomial order is increased. To demonstrate this on a scalar model problem, we consider the approximation of the advection-diffusion equation, (35) on $[0, 2] \times [0, 2]$. The diffusion coefficient was chosen to be $\nu = 0.5$, and the source term was chosen so that the exact steady-state

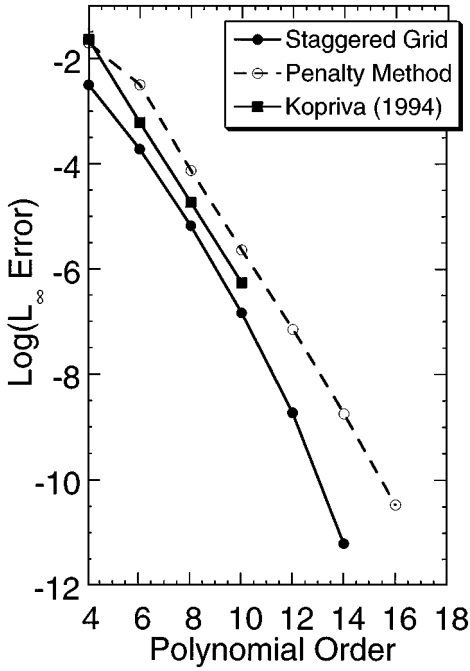


**FIG. 11.** Convergence of the error for the advective-diffusion equation in two space dimensions.

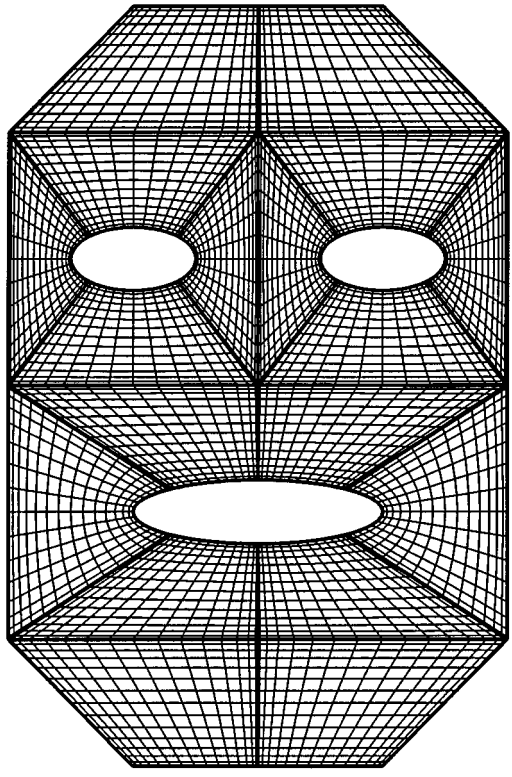

**FIG. 12.** Grid for the solution of the linear system of equations.

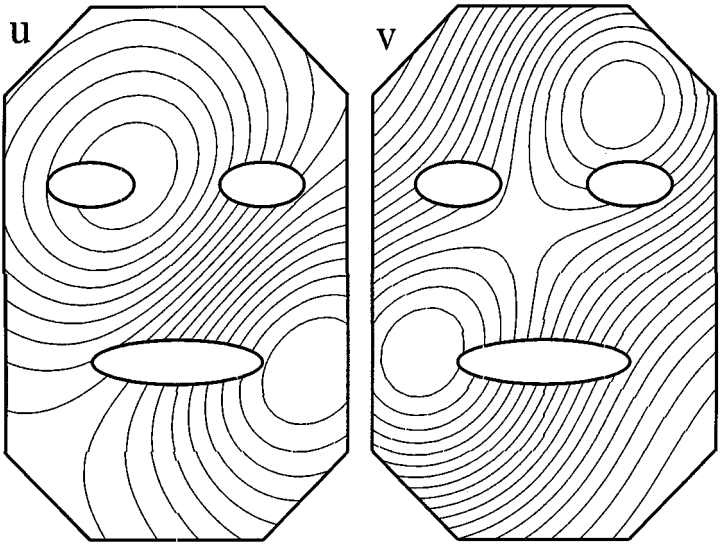


**FIG. 13.** Steady solutions $(u, v)$ to the linear system.

solution was

$$v = e^{-2[(x-x_1)^2+(y-y_1)^2]} + e^{-2[(x-x_2)^2+(y-y_2)^2]} \tag{39}$$

with $(x_1, y_1) = (0.9, 0.3)$, and $(x_1, y_1) = (1.1, 1.6)$.

We first consider the convergence of a two subdomain decomposition of the square with order refinement, as shown in Fig. 8. Figure 15 shows the convergence of the $L^2$ error

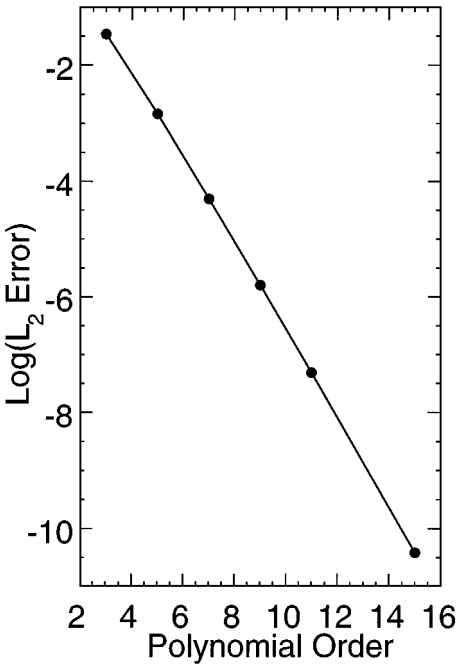


**FIG. 14.** Convergence of the $L^2$ error as a function of polynomial order for the linear system.

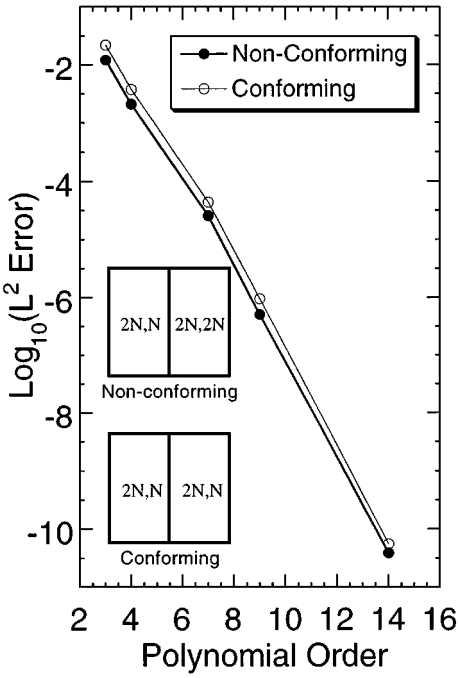


**FIG. 15.** Comparison of conforming and non-conforming errors under order refinement.

when twice as many vertical collocation points are used on the right as on the left. For this problem, local refinement is not needed, so we would expect the error to be no worse than a conforming approximation with the lower resolution. Figure 15 shows that this is indeed the case.

Spectral accuracy is also observed when subdomains are subdivided. In Fig. 16, we compare the error of an equal four-subdomain conforming decomposition of the square with a non-conforming subdivided decomposition. Again, we see that the error of the subdivided decomposition behaves like that of the conforming approximation, which indicates that the error is being dominated by the lowest order approximation. Plotted also on Fig. 16 is an example showing spectral error decay for a subdivided decomposition using the same order polynomials in each direction and in each subdomain.

### 4.2. Navier–Stokes Solutions

We now consider the solution of the Navier–Stokes equations. We first solve the Couette flow [25], which describes the flow between a moving and a fixed flat plate. This problem has an exact solution, so we can study the convergence of the staggered grid method on a non-linear problem. Next, we consider the flow over a subsonic flat plate. While there is no exact solution in the case of compressible flow, we compare our solution to that of a spectral boundary-layer approximation [24]. To show that the method is applicable to transonic flows, we apply it to a viscous flow in a converging-diverging nozzle. The computed results of this calculation are compared to experimental data [14]. Finally, we solve a time dependent subsonic flow over a cylinder and compare the computed vortex shedding frequency to experimental results.

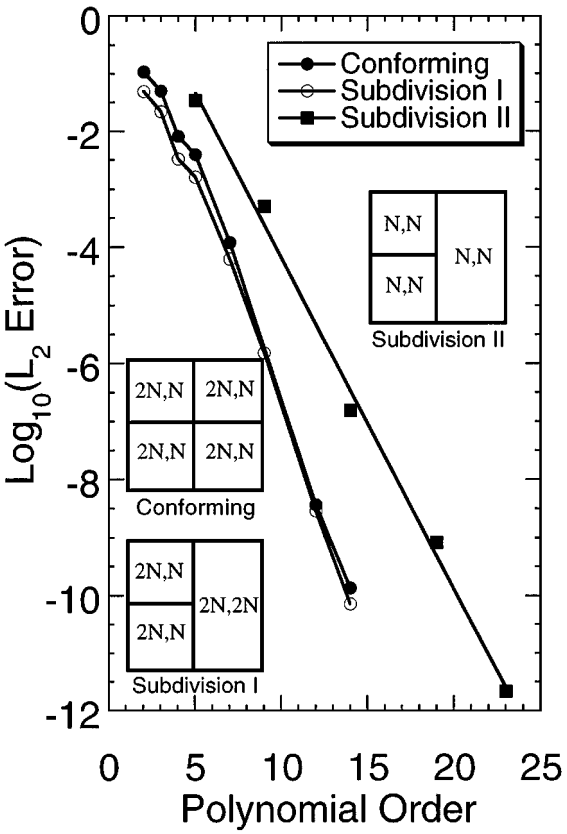


**FIG. 16.** Comparison of $L^2$ errors for subdivision of subdomains.

4.2.1. *The Couette Flow*

As the first example of the Navier–Stokes equations by the staggered-grid approximation, we present the solution of the Couette flow [25]. This problem models the viscous flow between a fixed, insulated lower plate, and a moving, fixed temperature, upper plate. It has an exact solution under the simplification that the viscosity coefficient $\mu = T$. We compute this flow in the unit square with an upper plate speed chosen so that the Mach number along the upper plate is $M = 0.8$ and $\mathrm{Pr} = 0.72$. We present results for two values of the Reynolds number: $\mathrm{Re} = 50$ and $\mathrm{Re} = 500$ (corresponding to $\mathrm{Re}_v = 47.3$ and $\mathrm{Re}_v = 473$). The upper plate temperature was set to one, and the adiabatic boundary condition was applied at the lower plate, according to the discussion above. Since the exact solution is known, exact values of the inflow and outflow quantities were used as necessary to compute the inviscid and viscous flux.

We solve the Couette flow on the unstructured quadrilateral grid shown in Fig. 17, which has nine subdomains. On this grid, the one-dimensional Couette flow becomes a two dimensional problem, as far as the numerical scheme is concerned. Convergence of the error in the density is exponential, as shown in Fig. 18.

4.2.2. *Subsonic Flow over a Flat Plate*

The second example is that of a steady subsonic flow over a thin insulated and a fixed temperature flat plate. The plate begins at $(x, y) = (0, 0)$ and extends to $(+\infty, 0)$. We solve

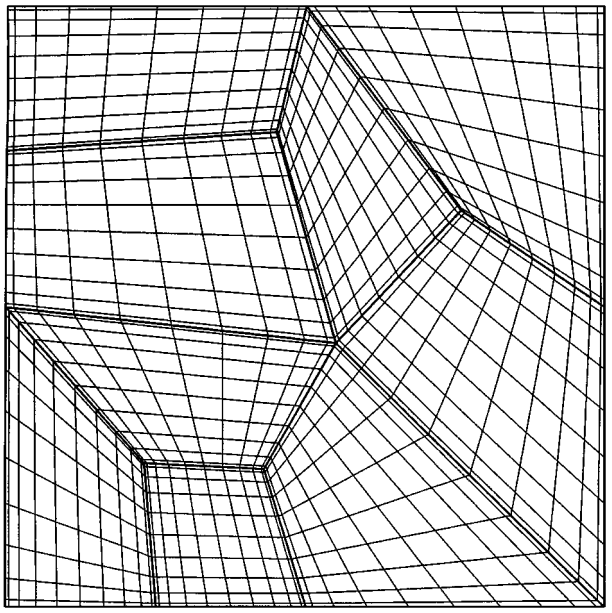

**FIG. 17.** Grid with nine subdomains for the Couette flow computation.

the problem on the rectangular domain $\Omega = [-16, 20] \times [0, 8]$, which includes the edge of the plate. A total of 55 subdomains were used. For the adiabatic plate, we have also used the domain $\Omega = [-35, 20] \times [0, 35]$ and 61 subdomains to reduce the effects of the upper boundary. Since the front edge of the plate is included in the domain, we use a non-conforming approximation with subdomain refinement rather than order refinement to resolve better the solution at the leading edge (cf. [21]). The grid shown in Fig. 19

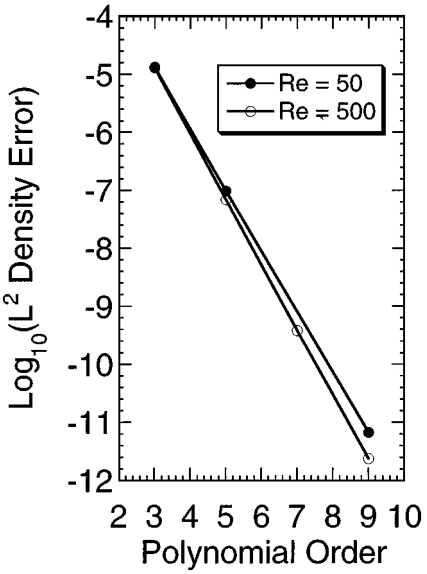


**FIG. 18.** Convergence of the density error for the Couette flow on the grid shown in Fig. 17.

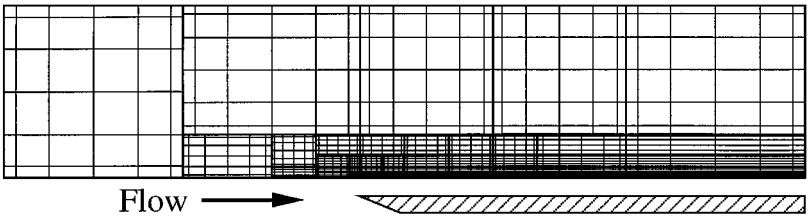


**FIG. 19.** Non-conforming grid used for the solution of the flat plate problem.

approximates the solution with a $5 \times 5$ order polynomial. The leading edge of the plate is placed at a subdomain interface.

The physical parameters of this flow are free-stream Mach number $M_\infty = 0.6$ and $\mathrm{Re}_v = 2500$ based on a unit length. The mean flow temperature was chosen to be $T_\infty = 520\mathrm{R}$. For the isothermal plate, we set the plate temperature to $T_w = 780\mathrm{R}$. At the left boundary, we set the mean flow values and $\tau_{xx} = 0$. Ahead of the plate along $y = 0$, a symmetry condition is set. This is accomplished by treating the boundary as a free-slip wall and enforcing the condition that solution derivatives be zero when computing the viscous flux. The top boundary is treated by setting the mean flow solution values in the Riemann solver for the inviscid flux, and by setting the viscous stresses to zero according to whether the flow is entering or exiting the boundary. Finally, the subsonic outflow boundary at the right is treated by setting the pressure, $p = 1$, and $T_x$, $\tau_{yx}$ to zero.

For initial conditions, we use an impulsive start, with free-stream conditions everywhere. The flow was then marched in time until the maximum residual in all variables dropped to rounding error levels. As an example, we show the convergence to rounding error of the maximum residual for the isothermal plate and seventh order polynomial approximations in Fig. 20. Solution contours for the steady Mach number and temperature for a portion of the grid near the plate edge are shown in Figs. 21 and 22.

There is no exact solution for the compressible boundary-layer problem. However, comparisons can be made to the solutions of a boundary layer approximation far enough from

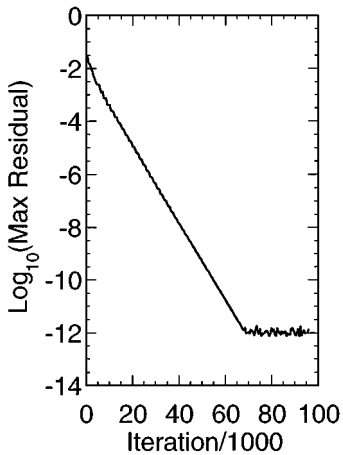


**FIG. 20.** Convergence to steady-state of the residual for the flow over a flat plate in a free-stream with $M_\infty = 0.6$.

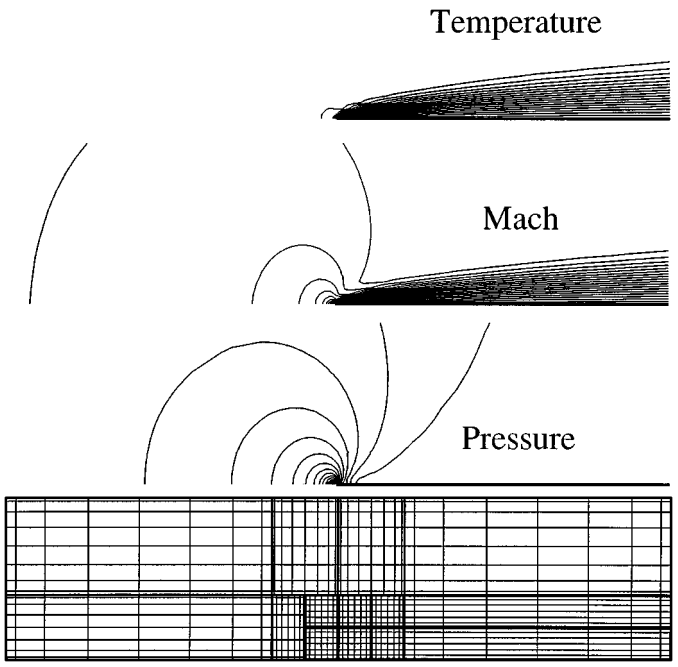


**FIG. 21.** Temperature and Mach number contours in the vicinity of the plate edge shown in Fig. 19 for the fixed temperature plate.

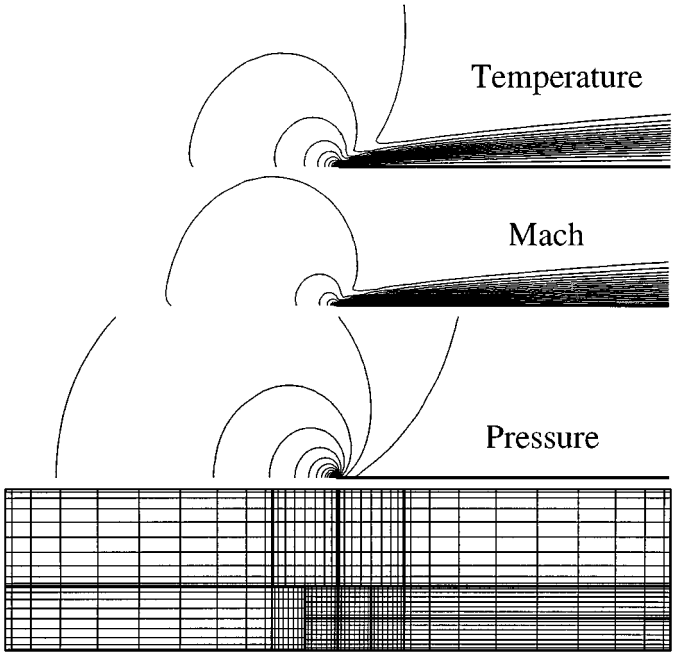


**FIG. 22.** Temperature and Mach number contours in the vicinity of the plate edge shown in Fig. 19 for the insulated plate.

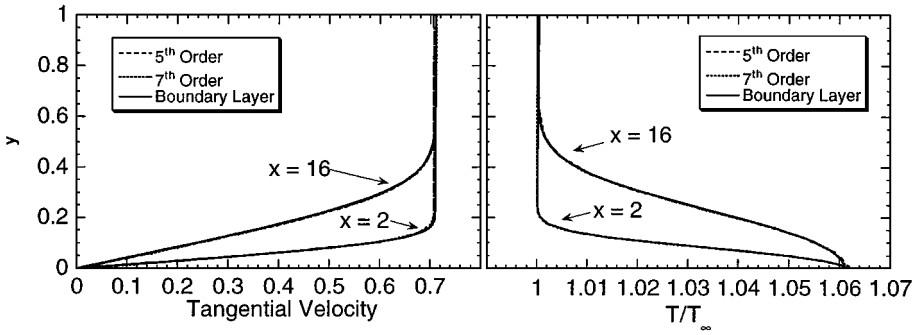


**FIG. 23.** Computed temperature and tangential velocity near the plate at $x = 1$ and $x = 16$ compared to a compressible boundary-layer solution for the insulated plate.

the plate edge. Figures 23 and 24 compare the boundary layer profiles at $x = 2$ and $x = 16$ to the solution computed by the boundary-layer code of [24].

What we see in Figs. 23 and 24 is that the solutions are grid-converged to graphical accuracy by $N = 9$. In the adiabatic case, the wall temperature is accurate to 0.04 and 0.03% at $x = 2$ and $x = 16$, respectively. The RMS free-stream temperatures are accurate to 0.04 and 0.05% at those $x$-stations, while the tangential velocity is accurate to 0.1 and 0.2%. The fifth order approximation, which has only 15 points across the boundary layer at $x = 16$, is not grid converged for the isothermal plate. However, the error in the RMS velocity and temperature in the free-stream of the ninth order approximation at $x = 2$ is 0.8 and 0.2%. At $x = 16$, the computed free-stream values are accurate to 0.03% in the temperature and 0.2% in the tangential velocity.

### 4.2.3. *Transonic Flow in a Converging-Diverging Nozzle*

As an example of a transonic problem, we consider the flow in a converging-diverging nozzle. The nozzle contour we use is the 5 × Configuration 1 chemical laser nozzle with a throat gap of 0.05 inch studied in [14]. The flow parameters given for the nozzle were a total temperature of 520°R, total pressure $p_{tot} = 1$ psia and Reynolds number $\mathrm{Re}_v = 1200$ based on the throat gap. We solve the flow for an isothermal wall, set to the stagnation

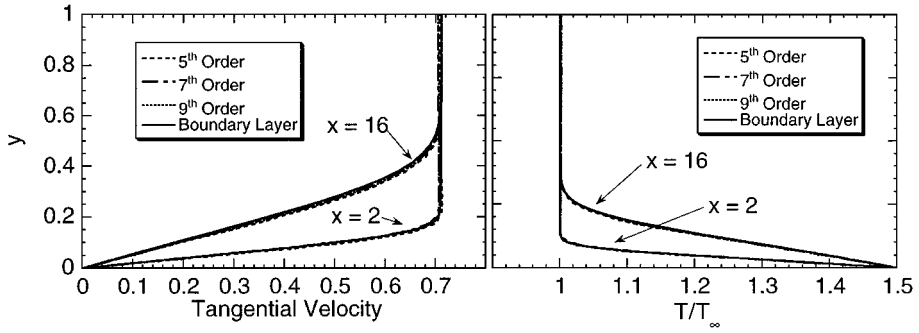


**FIG. 24.** Computed temperature and tangential velocity near the plate at $x = 1$ and $x = 16$ compared to a compressible boundary-layer solution for the fixed temperature plate.

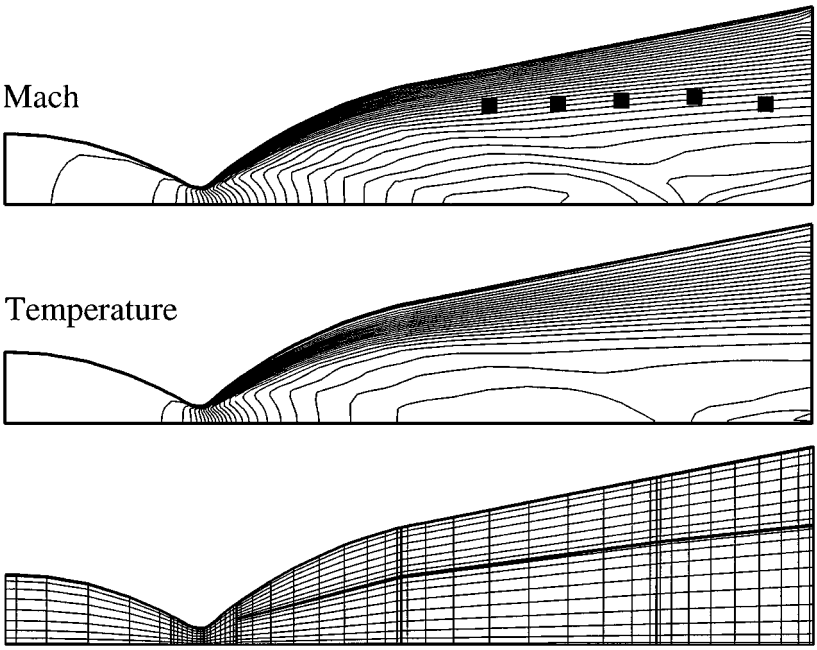


**FIG. 25.** Nozzle contours and grid. At the top are Mach contours showing the location of the boundary layer edge from the experiment.

temperature. In the boundary layer, the outflow pressure, $p/p_{tot}$ was set to the experimental value of 0.004.

The nozzle geometry and grid are shown on the bottom of Fig. 25. We solve for the flow in the upper half of the nozzle, and use a symmetry condition along the lower boundary. The domain is covered with 9 subdomains, chosen so that the throat and the boundary layer are sufficiently resolved. The quasi-one-dimensional inviscid solution as used to start the computation.

Contours of the Mach number and temperature are also shown in Fig. 25. Plotted with the Mach contours are the experimentally determined positions of the boundary layer. We see good visual agreement in the growth of the boundary layer between the experiment and the computed solution. Figure 26 shows the velocity scaled to the sound speed at the throat at a distance of 13 throat radii from the throat. This location corresponds to the last vertical subdomain boundary on Fig. 25. Again, agreement to within the scatter of the experimental data is observed.

### 4.2.4. *Subsonic Flow over a Circular Cylinder*

As our final example, we consider the subsonic flow over a circular cylinder. At Reynolds numbers between about 40 and 150 an unsteady, stable vortex street forms behind the cylinder. Measured values of the Strouhal number of the vortex shedding frequency can be found in [26]. Spectral solutions of cylinder wake flows include those of [6, 12]. A recent detailed review of the problem can be found in [3].

We compute the solution for Mach number 0.2 and $\mathrm{Re}_v = 75$ on a non-conforming grid. A portion of that grid in the neighborhood of the cylinder is shown in Fig. 27. The full grid extends to 20 cylinder diameters in each direction. To start the computation, we used

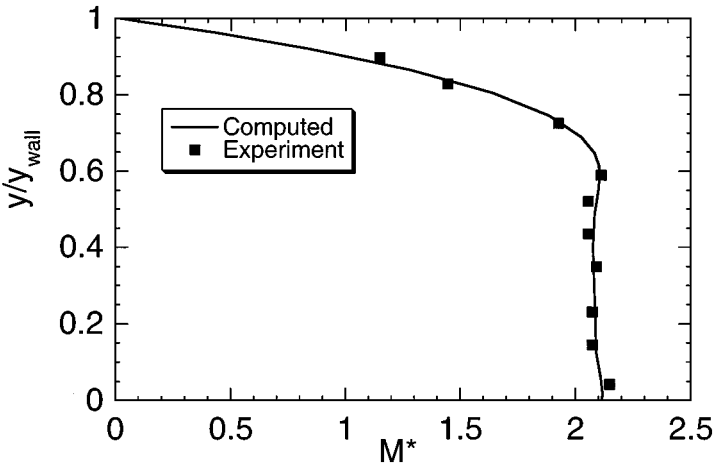


**FIG. 26.** Vertical variation of the velocity scaled to the throat sound speed at $x = 13$.

a uniform flow everywhere to which an upstream, off-centered pressure perturbation was added to break the symmetry.

After a sufficiently long time, the effects of the initial condition propagate out of the computational domain, and the periodic shedding of vorticies is observed. Instantaneous contours of the Mach number and vorticity showing the von Karman vortex street generated by the cylinder are presented in Figs. 28 and 29. The periodic nature of the flow is shown in Fig. 30, which plots as a function of time the pressure measured ten diameters downstream of the cylinder and one diameter up from the centerline. The period of the oscillations corresponds to a Strouhal number of 0.150, compared to the value of 0.149 reported in [12, 26].

## 5. SUMMARY

In this paper, we have presented a staggered-grid Chebyshev multidomain spectral method for the solution of the compressible Navier–Stokes equations. The method collocates the

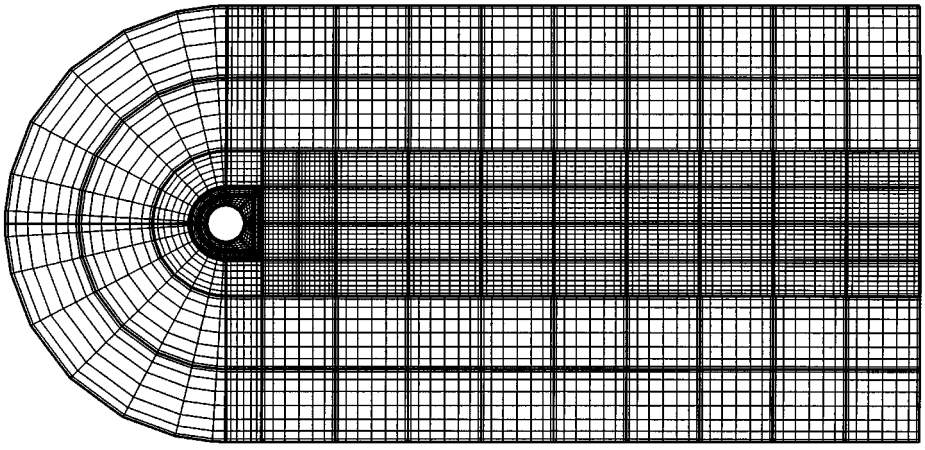

**FIG. 27.** Portion of the grid for the computation of the unsteady flow over a circular cylinder.

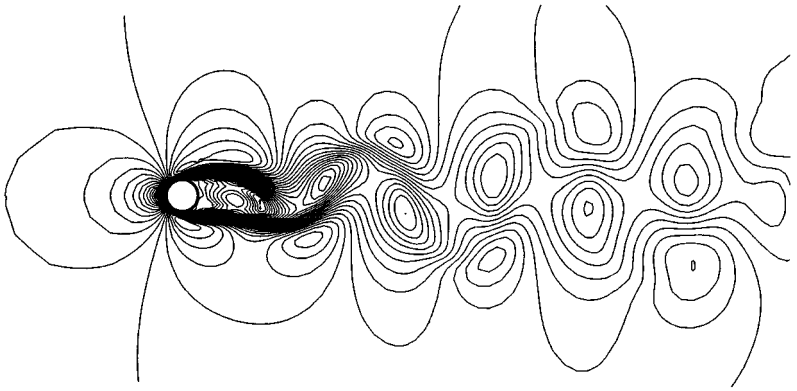

**FIG. 28.** Instantaneous Mach contours for $M = 0.2$ flow over a circular cylinder at $\mathrm{Re}_v = 75$.

solution unknowns and the gradients at the nodes of the Gauss–Chebyshev quadrature. The total fluxes are evaluated at the notes of the Gauss–Lobatto quadrature. At subdomain interfaces, the advective fluxes are computed by a Riemann solver to allow for the passage of waves. The viscous fluxes are computed by constructing a continuous piecewise polynomial approximation to the solution from the piecewise discontinuous approximation. Both conforming and non-conforming grids are allowed. In the non-conforming case, a least-squares approximation to a continuous flux is computed along a mortar that couples two subdomains.

Examples have been included to show the behavior of the method on both one- and two-dimensional linear and non-linear problems. Exponential convergence was shown on all problems for which exact solutions were known, for both conforming and non-conforming grid topologies. Examples of solutions of the full Navier–Stokes equations include the steady subsonic flow over a flat plate, steady transonic flow in a converging-diverging nozzle, and time dependent subsonic flow over a circular cylinder. For the problems on which they were compared, the method has accuracy similar to the penalty method recently presented in [11].

The method was designed with the goal of simplifying the connectivity between subdomains, and to eliminate the need for special coding at corner points of subdomains. As a result, complex quadrilateral grid topologies can be used an easily as simple ones, making the method easy to code and grids relatively easy to generate. This simplification should have even more of an effect in three-dimensional applications, where a subdomain will have at most six neighbors with which to communicate.

An added benefit of the discontinuous nature of the approximation at subdomain interfaces is robustness. Problems with discontinuities such as the flow over a flat plate can be computed

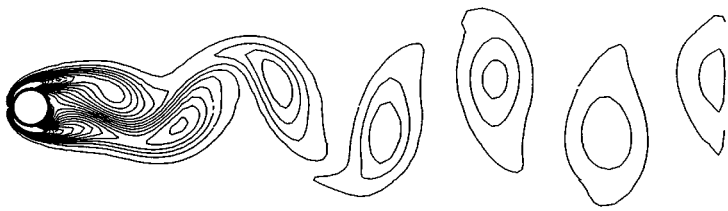

**FIG. 29.** Instantaneous vorticity contours for $M = 0.2$ flow over a circular cylinder at $\mathrm{Re}_v = 75$.

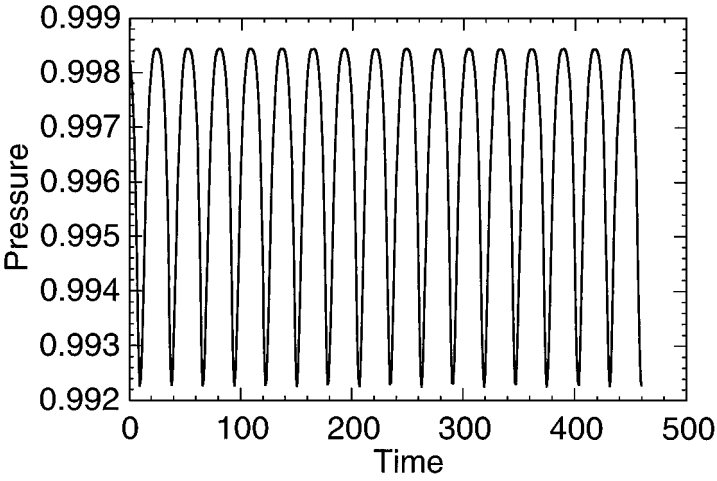


**FIG. 30.** Pressure trace at $(x, y) = (10, 1)$.

if the discontinuities are placed at interfaces without special coding, as with a cell centered finite volume method. Symmetry boundaries in axisymmetric flows are computed without special treatment. This contrasts with a Lobatto grid method where special corner and symmetry operators must be developed for every situation. Experience has also shown that the method is robust with respect to initial conditions. Impulsive start conditions that converge stably to steady-state with the staggered-grid approximation are generally unstable when continuity is required at subdomain interfaces.

The increased flexibility and robustness of the method comes, however, at increased computational cost. As in the case of the Euler equations [18], the number of matrix operations is formally twice that required by a non-staggered Lobatto grid method. This is because, in addition to the matrix operations for the derivatives, the staggered-grid approximation also requires matrix operations to project the solution values onto the flux grids. In our implementation, the matrix operations corresponding to differentiation and interpolation account for 46% for the total computational cost per time step. Based on this, we conclude that the staggered-grid approximation requires approximately 30% more computer time than one that uses only the Lobatto grid, not counting the extra work to compute special corner operators.

## ACKNOWLEDGMENTS

The author thanks Mr. J. Kolias, who helped in the early stages of this investigation. He also thanks Dr. J. S. Hesthaven for useful criticisms and for supplying his results for comparison. Finally, thanks to Dr. M. C. Cline for supplying the nozzle profile.

## REFERENCES

1. C. Basdevant, M. Deville, P. Haldenwang, J. M. Lacroix, J. Ouazzani, R. Peyret, P. Orlandi, and A. T. Patera, Spectral and finite difference solutions of the Burgers equation, *Comput. & Fluids* **14**, 23 (1986).
2. F. Bassi and S. Rebay, A high order accurate discontinuous finite element method for the numerical solution of the compressible Navier–Stokes equations, *J. Comput. Phys.* **131**, 267 (1997).
3. P. Beaudan and P. Moin, *Numerical Experiments on the Flow Past a Circular Cylinder at Sub-critical Reynolds Number*, Technical Report, TF-62, Department of Mech. E., Stanford University, 1994.

4. C. Canuto, M. Y. Hussaini, A. Quarteroni, and T. A. Zang, *Spectral Methods in Fluid Dynamics* (Springer-Verlag, New York, 1987).
5. M. Carpenter and C. Kennedy, Fourth-order 2N-storage Runge–Kutta schemes, NASA TM 109112, 1994, unpublished.
6. W.-S. Don and D. Gottlieb, Spectral simulation of unsteady compressible flow past a circular cylinder, *Comput. Methods Appl. Mech. Eng.* **80**, 39 (1990).
7. P. Dutt, Stable boundary conditions and difference schemes for Navier–Stokes equations, *SIAM J. Numer. Anal.* **25**, 245 (1988).
8. J. Giannakouros, *Spectral Element/Flux-Corrected Methods for Unsteady Compressible Viscous Flows*, Ph.D. thesis, Princeton University, Princeton, NJ, 1994.
9. B. Gustaffson and A. Sundstrom, Incompletely parabolic problems in fluid dynamics, *SIAM J. Appl. Math.* **35**, 343 (1978).
10. P. Hanley, A strategy for the efficient simulation of viscous compressible flows using a multi-domain pseudo-spectral method, *J. Comput. Phys.* **108**, 153 (1993).
11. J. S. Hesthaven, A stable penalty method for the compressible Navier–Stokes equations. iii. Multidimensional domain decomposition schemes, *SIAM J. Sci. Comput.*, in press.
12. J. S. Hesthaven, A stable penalty method for the compressible Navier–Stokes equations. ii. One dimensional domain decomposition schemes, *SIAM J. Sci. Comput.* **18**, 658 (1997).
13. J. S. Hesthaven and D. Gottlieb, A stable penalty method for the compressible Navier–Stokes equations. i. Open boundary conditions, *SIAM J. Sci. Comput.* **17**, 579 (1996).
14. J. C. Hyde and G. A. Hosack, *An Investigation of Velocity Flowfields in Chemical Laser Nozzles*, AIAA-Paper 73-641, 1973.
15. D. A. Kopriva, A spectral multidomain method for the Euler gas-dynamics equations, *J. Comput. Phys.* **96**, 428 (1991).
16. D. A. Kopriva, Multidomain spectral solution of compressible viscous flows, *J. Comput. Phys.* **115**, 184 (1994).
17. D. A. Kopriva, A conservative staggered-grid chebyshev multidomain method for compressible flows. ii. A semi-structured method, *J. Comput. Phys.* **128**, 475 (1996).
18. D. A. Kopriva and J. H. Kolias, A conservative staggered-grid chebyshev multidomain method for compressible flows, *J. Comput. Phys.* **125**, 244 (1996).
19. I. Lomtev, C. W. Quillen, and G. Karniadakis, *Spectral/hp Methods for Viscous Compressible Flows on Unstructured 2d Meshes*, Technical Report 96-12, Center for Fluid Mechanics Turbulence and Computation, Brown University, 1996.
20. M. Macaraeg and C. L. Streett, A spectral multi-domain technique for viscous compressible reacting flows, *Int. J. Numer. Methods Fluids* **8**, 1121 (1988).
21. C. A. Mavriplis, *Nonconforming Discretizations and a Posteriori Error Estimates for Adaptive Spectral Element Techniques*, Ph.D. thesis, MIT, Cambridge, MA, 1989.
22. J. Oliger and A. Sundstrom, Theoretical and practical aspects of some initial boundary value problems in fluid dynamics, *SIAM J. Appl. Math.* **35**, 419 (1978).
23. T. J. Pointsot and S. K. Lele, Boundary conditions for direct simulations of compressible viscous reacting flows, *J. Comput. Phys.* **101**, 104 (1992).
24. C. D. Pruett and C. L. Streett, A spectral collocation method for compressible, non-similar boundary layers, *Int. J. Numer. Methods Fluids* **13**, 713 (1991).
25. F. M. White, *Viscous Fluid Flow* (McGraw–Hill, New York, 1974).
26. C. H. K. Williamson, Oblique and parallel modes of vortex shedding in the wake of a cylinder at low Reynolds number, *J. Fluid Mech.* **206**, 579 (1989).
27. J. H. Williamson, Low storage Runge–Kutta schemes, *J. Comput. Phys.* **35**, 48 (1980).
28. T. A. Zang and M. Y. Hussaini, Mixed spectral/finite difference approximations for slightly viscous flows, in *Seventh International Conference on Numerical Methods in Fluid Dynamics*, edited by W. C. Reynolds and R. W. MacCormack, Lect. Notes in Phys. (Springer-Verlag, New York/Berlin, 1981), Vol. 141.