JOURNAL OF COMPUTATIONAL PHYSICS **125,** 135–149 (1996)
ARTICLE NO. 0084

A New Efficient Algorithm for Computational Aeroacoustics on Parallel Processors

YUSUF ÖZYÖRÜK AND LYLE N. LONG

Department of Aerospace Engineering, Pennsylvania State University, University Park, Pennsylvania 16802

Received June 2, 1995; revised November 1995

One of the great challenges in computational physics is the prediction of flow associated noise, where the quantities of interest, namely the sound waves can be at high frequencies and are usually orders of magnitude smaller in magnitude than the mean quantities. In order to numerically resolve such small scales governed by the fluid dynamics equations, high resolution schemes are required. Thus solutions of flow noise problems are computationally intensive. An efficient, hybrid, data parallel computational aeroacoustics algorithm has been developed for the prediction of noise radiation and scattering from three-dimensional geometries. The algorithm solves the Euler/Navier–Stokes equations in the interior and nonreflecting boundary conditions on the outer boundaries. A moving surface Kirchhoff method is coupled to the flow solver for far-field predictions. The algorithm uses standard time and spatial discretization techniques but utilizes several new optimization strategies that are highly suitable for single zone solutions on data parallel processors. One strategy, for example, enables simultaneous residual evaluations of the interior and far-field nonreflecting boundary conditions equations, reducing the computational effort spent on them by approximately 60% CPU time savings. The algorithms for the flow solver and the Kirchhoff method and their coupling are described in this paper, and results for some example radiation and scattering problems are presented.

1. INTRODUCTION

The prediction of flow associated noise is one of the great challenges in computational physics. Two examples of such problems are jet noise [1–3] and turbofan noise [4–6]. Turbulence and large structure instability waves are often responsible for the production of noise in a jet, while rotor–stator interactions are responsible for most of the fan noise. The physics and the noise generation mechanisms may vary greatly from one problem to another, but the far-field predictions all involve propagating waves numerically over long distances and long time periods. Quantities of interest may be at high frequencies and are usually orders of magnitude smaller than the mean quantities. The solution of the problems involving such small scales requires high order accurate numerical algorithms, which have relatively good dissipation and dispersion characteristics. Due to computational cost limitations, calculations are carried out in finite domains with nonreflecting boundary conditions. These conditions, which represent additional time-dependent partial differential equations, are extremely important for successful aeroacoustic simulations. Thus computational techniques are challenged by the complexity of the physics, and the computational power is challenged by the required fine grids and large number of computations.

Parallel computers offer high speeds and massive amounts of memory, which make them attractive for computational aeroacoustic problems, where one has to solve different sets of time dependent partial differential equations; radiation boundary conditions on the inflow boundaries, outflow boundary conditions on the outflow boundaries [7, 8], and the Euler/Navier–Stokes equations in the interior. Efficient handling of the outer boundaries is a very difficult task when single zone solutions are performed in the data parallel mode on computers such as the Connection Machine-5 (CM-5). In the data parallel mode one solves for the interior points and the outer boundary points in a sequential manner. This, however, means at least twice as much CPU time in differentiating the fluxes of the governing equations. In addition, floating point operations are relatively inexpensive, while communication among the processors can be quite time consuming.

Communication cost is more pronounced for high order accurate schemes because they necessitate relatively large stencils. One could cut down the communication time by a factor of $(N - 1)/N$ through a selection process of the appropriate variables to be differentiated depending upon the location on the mesh, where N is the number of the different sets of governing equations. This new method utilizes the `MERGE` statement that the programming languages CM Fortran [9], Fortran 90, and high performance Fortran (HPF) [10] facilitate. This command basically combines the elements of two different argument arrays (arrays of the quantities to be differentiated in this paper) into a new array under a given logical masking array.

This paper describes a hybrid, data parallel computational aeroacoustics algorithm that has been developed to solve noise radiation and scattering from three dimensional geometries. The algorithm solves the 3D Navier–Stokes/

0021-9991/96 $18.00

Euler equations, together with time-dependent nonreflecting boundary conditions using a spatially and temporally fourth-order accurate finite difference, Runge–Kutta (R-K) time integration scheme. The governing equations and nonreflecting boundary conditions are formulated in either the Cartesian or cylindrical coordinates with a switching parameter and then transformed into a 3D body-fitted curvilinear coordinate system. Formulations in cylindrical coordinates allow proper treatment of the grid singularity in some problems, such as engine inlet flow and acoustics. Unlike the conventional methods [11], central differencing is used across the centerline of an inlet.

The flow solver is coupled to a far-field acoustics extrapolation code based on the moving surface Kirchhoff formula of Farassat and Myers [12]. The Kirchhoff method is a surface integral method that enables one to limit the size of the computational fluid dynamics (CFD) mesh so that the accumulation of numerical errors (dissipation and dispersion) can be prevented and more importantly the far-field noise can be obtained with minimal computational resources.

Including a simultaneous residual evaluation procedure, several other new optimization techniques are utilized in the computer code. In the following sections, the numerical algorithm for the flow solver, the optimization strategies, and the coupling of the flow solver with the Kirchhoff method are described. Then some example results for both steady and time accurate calculations are presented, and finally some conclusions are drawn.

2. NUMERICAL ALGORITHM

2.1. *Governing Equations*

The Navier–Stokes/Euler equations are integrated in the interior of the domain, together with nonreflecting boundary conditions on the outer boundaries of the computational mesh.

The far-field boundaries are in general composed of inflow and outflow boundaries, which require different conditions depending on the characteristics. At a subsonic inflow boundary, the only characteristic that is out-going is associated with the acoustic waves and, therefore, radiation boundary conditions based on the B_1 operator of Bayliss and Turkel [7] are applied, which is accurate asymptotically to $O(d^{-3})$, where d is the Prandtl–Glauert transformed spherical distance of the boundary point from a reference point, which is usually taken to be the approximate source location. Bayliss and Turkel constructed a family of partial differential operators B_{m} by using the asymptotic solution of the convective wave equation. Tam and Webb [8] derived radiation and outflow boundary conditions based on the asymptotic solution of the linearized Euler equations. Their radiation boundary condition operator and that of Bayliss and Turkel are essentially the same.

One can show that in the absence of vorticity the perturbed velocity components satisfy the convective wave equation, as well as the density and pressure perturbations, and, therefore, in the current numerical scheme the B_1 operator is applied to all the primitive perturbation flow quantities on the subsonic inflow boundaries. This is exactly equivalent to the radiation boundary conditions of Tam and Webb [8].

At a subsonic outflow boundary, the characteristics that are out-going are associated with acoustic waves, entropy waves, and vorticity waves. Therefore, at such boundaries the B_1 operator is applied only to the pressure perturbation, while the other perturbed variables are obtained by solving the linearized Euler equations, following Tam and Webb [8].

In general the interior governing equations and the far-field governing equations with a choice of the Cartesian or cylindrical coordinate system with the parameter m are all given by

$$\frac{\partial \mathbf{Q}}{\partial t} + \frac{\partial \mathbf{E}}{\partial x_1} + \frac{\partial \mathbf{F}}{\partial x_2} + \frac{1}{x_2^m}\frac{\partial \mathbf{G}}{\partial x_3} + \frac{\mathbf{S}_B + m\mathbf{S}_C}{x_2^m} = 0, \tag{1}$$

where $\mathbf{Q}$ is the vector of dependent variables and $\mathbf{E}$, $\mathbf{F}$, and $\mathbf{G}$ are the vectors which are functions of $\mathbf{Q}$ (flux vector components for interior points) in the x_1, x_2, and x_3 coordinate directions, respectively. In the above equation the parameter $m = 0$ for the Cartesian coordinate system $(x_1, x_2, x_3) = (x, y, z)$, $m = 1$ for the cylindrical coordinate system $(x_1, x_2, x_3) = (x, r, \theta)$, and the vectors $\mathbf{S}_B$ and $\mathbf{S}_C$ are the source terms which are associated with the B_1 operator of the far-field governing equations and the equations in cylindrical coordinates, respectively. In general, for the dependent variables $\mathbf{Q}$ and the functions $\mathbf{E}$, $\mathbf{F}$, $\mathbf{G}$, and $\mathbf{S}$ we write

$$\{\mathbf{Q}, \mathbf{E}, \mathbf{F}, \mathbf{G}, \mathbf{S}\} = \begin{cases} \{\mathbf{Q}, \mathbf{E}, \mathbf{F}, \mathbf{G}, \mathbf{S}\}_{\mathrm{int}} & \mathrm{N-S/Euler\ Eqs.}, \\ \{\mathbf{Q}, \mathbf{E}, \mathbf{F}, \mathbf{G}, \mathbf{S}\}_{\mathrm{rad}} & \mathrm{Radiation\ BCs}, \\ \{\mathbf{Q}, \mathbf{E}, \mathbf{F}, \mathbf{G}, \mathbf{S}\}_{\mathrm{out}} & \mathrm{Outflow\ BCs}. \end{cases} \tag{2}$$

The vector of state variables for the interior points is $\mathbf{Q}_{\mathrm{int}} = [\rho, \rho u_1, \rho u_2, \rho u_3, \rho e]^{\mathrm{T}}$ and for the far-field points $\mathbf{Q}_{\mathrm{rad,out}} = [\rho', u_1', u_2', u_3', p']^{\mathrm{T}}$, where ρ is the density; u_1, u_2, u_3 are the velocity components in the x_1, x_2, and x_3 coordinate directions, respectively; ρe is the total energy; and p is the pressure; and a prime indicates the deviation from the undisturbed (free stream) value of the associated quantity. The pressure is given by the equation of state as $p = (\gamma - 1)[\rho e - \frac{1}{2}\rho(u_1^2 + u_2^2 + u_3^2)]$. The terms $\mathbf{E}$, $\mathbf{F}$, $\mathbf{G}$, $\mathbf{S}$ for the Navier–Stokes/Euler equations can be found in many stan-

dard fluid dynamics textbooks and are not given here, but these terms for the far-field boundary conditions are given in the Appendix in a generalized form for arbitrary three-dimensional problems.

Equation (1) is transformed into a general 3D body-fitted curvilinear coordinate system, assuming

$$x_1 = x_1(\xi, \eta, \zeta), \quad x_2 = x_2(\xi, \eta, \zeta), \quad x_3 = x_3(\xi, \eta, \zeta). \tag{3}$$

Then,

$$\frac{\partial}{\partial t}\frac{\mathbf{Q}}{J} + \frac{\partial \tilde{\mathbf{E}}}{\partial \xi} + \frac{\partial \tilde{\mathbf{F}}}{\partial \eta} + \frac{\partial \tilde{\mathbf{G}}}{\partial \zeta} + \frac{\mathbf{S}_B + m\mathbf{S}_C}{Jx_2^m} = 0, \tag{4}$$

where

$$\begin{Bmatrix} \tilde{\mathbf{E}} \\ \tilde{\mathbf{F}} \\ \tilde{\mathbf{G}} \end{Bmatrix} = J^{-1} \begin{bmatrix} \xi_{x_1} & \xi_{x_2} & x_2^{-m}\xi_{x_3} \\ \eta_{x_1} & \eta_{x_2} & x_2^{-m}\eta_{x_3} \\ \zeta_{x_1} & \zeta_{x_2} & x_2^{-m}\zeta_{x_3} \end{bmatrix} \begin{Bmatrix} \mathbf{E} \\ \mathbf{F} \\ \mathbf{G} \end{Bmatrix}, \tag{5}$$

and J is the Jacobian of the coordinate transformation and is given by

$$J^{-1} = \left| \frac{\partial(x_1, x_2, x_3)}{\partial(\xi, \eta, \zeta)} \right|, \tag{6}$$

and the metric quantities are given by

$$\begin{bmatrix} \xi_{x_1} & \xi_{x_2} & \xi_{x_3} \\ \eta_{x_1} & \eta_{x_2} & \eta_{x_3} \\ \zeta_{x_1} & \zeta_{x_2} & \zeta_{x_3} \end{bmatrix} = \begin{bmatrix} x_{1_\xi} & x_{1_\eta} & x_{1_\zeta} \\ x_{2_\xi} & x_{2_\eta} & x_{2_\zeta} \\ x_{3_\xi} & x_{3_\eta} & x_{3_\zeta} \end{bmatrix}^{-1}. \tag{7}$$

Using the formulations given in cylindrical coordinates over those given in the Cartesian coordinates is critical, in terms of transforming the grid singularity encountered in meshes around bodies of revolution, such as the engine inlet, into only source-like terms ($\mathbf{S}_C$ above), thereby avoiding a possible violation of the geometrical conservation law during the discretization of Eq. (4).

2.2. *Grid System*

The numerical algorithm uses cell-centered finite differences. A second grid system is constructed from the cell centers of a given grid system, which are computed via third-order interpolations to preserve the smoothness of the original grid. The dependent variables are stored at the new grid points (see Fig. 1). Thus the possibility of having singular points (zero or negative Jacobians) is avoided. Across a singular line (e.g., the centerline in case of an engine inlet) the domain is treated as continuous, and formulations in cylindrical coordinates are used. This is important in terms of being able to use central differencing with no alteration of the cell sizes near the singular line, which, in time accurate calculations, are usually the dictating factors for the time step size of the integration scheme. However, the grid is required to be well constructed in the vicinity of the centerline, as well as in the other regions. In the ξ, η, and ζ coordinates the divergence of the inverse of the Jacobian times each column vector of the matrix in Eq. (5) should at least remain on the order of the accuracy of the spatial discretization (ideally zero, geometrical conservation law [13]). Even if the grid is orthogonal and treated as continuous across the singular line, this requirement is not satisfied in the Cartesian system, but it is satisfied with the metrics of this paper in cylindrical coordinates. Therefore, cylindrical coordinates are used in problems involving singular lines, and excellent results are obtained, as will be seen later on.

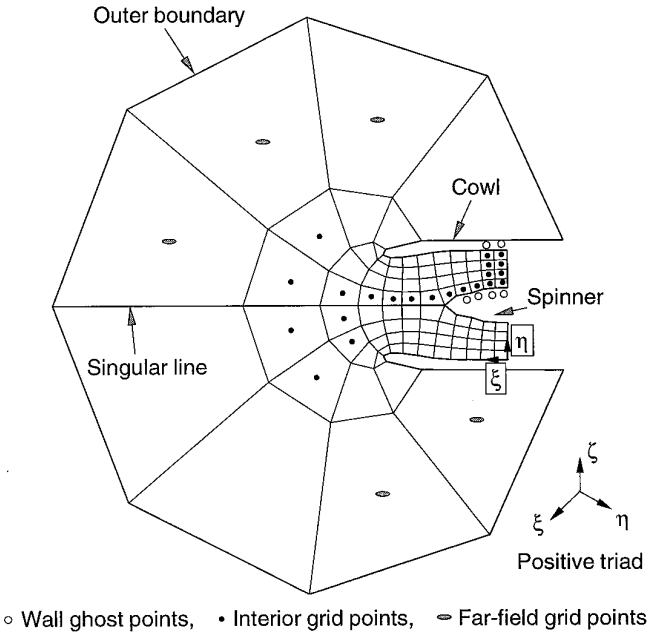

FIG. 1. Schematic of a typical grid around an engine inlet.

In order to apply the solid wall boundary conditions properly, ghost grid points are introduced inside the wall, such that in the mapped computational domain (ξ, η, ζ) the wall is at an equidistant position between the wall ghost point and the first grid point off the wall (see, for example, Fig. 1). Additional ghost cells along other boundaries are also utilized, and the grid points are re-indexed to implement the optimization techniques described in Section 2.4.

2.3. *Time Integration*

Equation (4) is integrated using the classical 4-stage R-K scheme. This scheme is fourth-order accurate in time and well proven for its effectiveness in the calculation of time dependent problems [14]. The stages of the R-K technique are given as

$$\begin{aligned}
\mathbf{Q}^{(1)} &= \mathbf{Q}^n, \\
\mathbf{Q}^{(2)} &= \mathbf{Q}^n - \tfrac{1}{2} J\,\Delta t[\mathscr{T}(\mathbf{Q}^{(1)}) - \mathscr{D}(\mathbf{Q}^{(1)})], \\
\mathbf{Q}^{(3)} &= \mathbf{Q}^n - \tfrac{1}{2} J\,\Delta t[\mathscr{T}(\mathbf{Q}^{(2)}) - \mathscr{D}(\mathbf{Q}^{(1)})], \\
\mathbf{Q}^{(4)} &= \mathbf{Q}^n - J\,\Delta t[\mathscr{T}(\mathbf{Q}^{(3)}) - \mathscr{D}(\mathbf{Q}^{(1)})], \\
\mathbf{Q}^{n+1} &= \mathbf{Q}^n - J\,\Delta t \sum_{s=1}^{4} \beta_s[\mathscr{T}(\mathbf{Q}^{(s)}) - \mathscr{D}(\mathbf{Q}^{(1)})],
\end{aligned} \tag{8}$$

where the superscript n shows the time step, Δt is the time increment from one step to the next, $\beta = [\frac{1}{6}, \frac{1}{3}, \frac{1}{3}, \frac{1}{6}]$, $\mathscr{T}$ is the residual vector, i.e., the sum $\partial\tilde{\mathbf{E}}/\partial\xi + \partial\tilde{\mathbf{F}}/\partial\eta + \partial\tilde{\mathbf{G}}/\partial\zeta + (\mathbf{S}_B + m\mathbf{S}_C)/Jx_2^m$, and $\mathscr{D}$ is the total artificial dissipation vector corresponding to variables $\mathbf{Q}$. Dissipation is necessary in central difference schemes because they lack mechanisms to suppress high frequency spurious oscillations. In most problems it is sufficient to calculate dissipation only once and freeze its value throughout all the stages. The R-K integration is stable for Courant numbers CFL $\leq 2\sqrt{2}$ when the spatial discretization is ignored. Von Neumann stability analysis applied to the 1D advection equation indicates that the fourth-order spatial discretization lowers this limit to about 2.06 [15, 16].

2.4. *Spatial Discretization and Parallelization*

Although finite volume methods are more robust in terms of representing the discretized version of the integral form of the conservation laws of fluid dynamics, higher order implementations become very expensive for three-dimensional problems due to required interpolation procedures [17]. Therefore, the spatial derivatives of Eq. (4) are computed using fourth-order accurate Taylor series derived finite differencing, which requires a 5-point stencil in each direction for every grid point. This stencil is central in the interior and biased near the boundaries.

The differences between the stencils of the interior grid points and the boundary or the near-boundary points cause difficulties in evaluating the derivatives in parallel. One approach to evaluate a derivative simultaneously at every grid point on a regular computational mesh within a single CM Fortran [9] line (so that there are no idle processors) is to use a 9-point stencil. Then at any point (i, j, k), the derivative $\partial\tilde{\mathbf{E}}/\partial\xi$ can be approximated with

$$\left.\frac{\partial\tilde{\mathbf{E}}}{\partial\xi}\right|_{i,j,k} = \frac{1}{\Delta\xi}\sum_{r=-4}^{4} a_{i,j,k}^{(r)}\tilde{\mathbf{E}}_{i+r,j,k}, \tag{9}$$

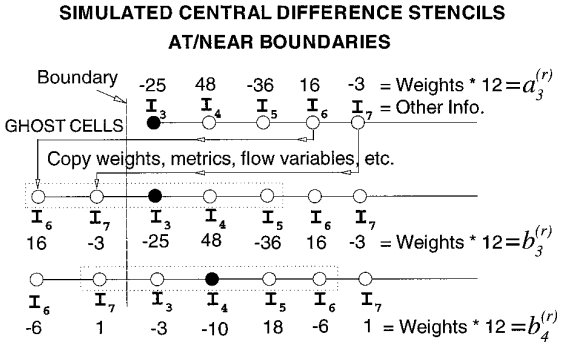

FIG. 2. Simulated central difference stencils at/near a boundary.

where $a_{i,j,k}^{(r)}$ are the weights (stencil arrays) and they take appropriate values such that the above sum in effect simulates a regular 5-point stencil for fourth-order spatial accuracy. However, this approach requires more storage for coefficients $a_{i,j,k}^{(r)}$ due to the integer r stretching from -4 to 4, and consequently more communication. This approach has been taken by some researchers (for example, see Refs. [18, 19]). The current paper uses a new approach, which is described below.

Two rows of ghost cells (grid points) are introduced along all the boundaries (solid wall, far-field, etc.). Then the first and the last real grid points in the domain are denoted by 3 and $N - 2$, respectively, where N is the total number of grid points (including ghost points) in the particular direction (ξ, η, or ζ). The ghost grid points are then loaded with the appropriate information (metrics and weights once, flow state variables at every Runge–Kutta stage, etc.) so that a simulated central difference stencil can be used. For example, Fig. 2 illustrates two simulated central difference stencils at points $i = 3$ (boundary point, first real cell) and $i = 4$ (one cell off the boundary, second real cell), respectively. This boundary has two imaginary cells to its left, which receive information from cells $i = 6$ and $i = 7$ so that for cell $i = 3$, cells $i = 1, 2, 3, 4$, and 5 establish a central difference stencil for the $\partial/\partial\xi$ derivative with weights $b_{i=3,j,k}^{(r)}$, r now extending from -2 to 2. And similarly for cell $i = 4$, cells $i = 2, 3, 4, 5$, and 6 with weights $b_{i=4,j,k}^{(r)}$ establish a central difference stencil. Thus for a derivative $\partial\tilde{\mathbf{E}}/\partial\xi$ at any point (i, j, k), it can be written that

$$\left.\frac{\partial\tilde{\mathbf{E}}}{\partial\xi}\right|_{i,j,k} = \sum_{r=-2}^{2} b_{i,j,k}^{(r)}\tilde{\mathbf{E}}_{i+r,j,k}. \tag{10}$$

Although introducing ghost cells at the boundaries may seem to increase the number of total grid points, this increase is only of $O(N^2)$, while Eq. (9) requires four additional stencil arrays of $O(N^3)$ for each direction, where N is the number of grid points in only one direction, as indicated

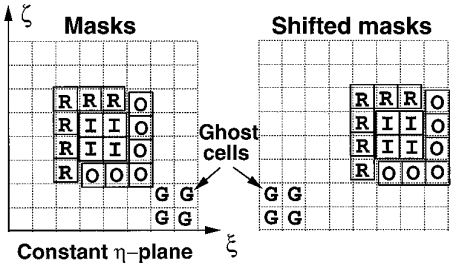

FIG. 3. Computational domain and the preshifted-by-two masks.

above. Also, the communication cost for loading the ghost cells with the appropriate information can be justified when compared to the unnecessary communication that is required by Eq. (9) for the apparent increased stencil size. Thus using this new approach (Eq. (10)) with the ghost cells at the boundaries, a given derivative can be evaluated simultaneously at every grid point in a more efficient way.

In solving an aeroacoustics problem on a data parallel computer (in SIMD mode), another difficulty, which is of real concern here, is the parallel evaluations of the residuals of the interior and the far-field boundary equations. This difficulty arises due to the fact that vector components **E**, **F**, and **G** of the flux vector tensor of the interior equations have different functional forms from their counterparts of the far-field inflow and outflow boundary condition equations (see Appendix). Thus the parallel evaluation of the derivatives of these vectors is a significant task. Here we develop a new programming strategy in CM Fortran or HPF to accomplish this task. The method makes use of the `MERGE` statement that these programming languages offer.

Consider a computational domain that has a far-field boundary on which the radiation and outflow boundary conditions are to be applied, and in the interior N-S equations are to be solved. A sketch of such a domain is illustrated in Fig. 3. The computational domain has ghost cells along the far-field boundaries for making the boundary stencils central as explained above. The radiation cells are denoted by R, the outflow by O, and the N-S by I. Our goal here, in a data parallel code, is to evaluate each of $\partial\tilde{\mathbf{E}}/\partial\xi$, $\partial\tilde{\mathbf{F}}/\partial\eta$, and $\partial\tilde{\mathbf{G}}/\partial\zeta$, which appear in Eq. (4), simultaneously everywhere in the computational domain.

Now consider the derivative $\partial\tilde{\mathbf{E}}/\partial\xi$ at the grid point (i, j, k). This derivative is given by Eq. (10), or for all the regions in an expanded form,

$$\left.\frac{\partial\tilde{\mathbf{E}}^c}{\partial\xi}\right|_{i,j,k} = (b_i^{(-2)}\tilde{\mathbf{E}}^c_{i-2} + b_i^{(-1)}\tilde{\mathbf{E}}^c_{i-1} + b_i^{(0)}\tilde{\mathbf{E}}^c_i + b_i^{(+1)}\tilde{\mathbf{E}}^c_{i+1} + b_i^{(+2)}\tilde{\mathbf{E}}^c_{i+2})_{,j,k}, \tag{11}$$

or equivalently in HPF,

$$\begin{aligned}\left.\frac{\partial\tilde{\mathbf{E}}^c}{\partial\xi}\right|_{i,j,k} &= b^{(-2)} * \texttt{cshift}(\tilde{\mathbf{E}}^c,\texttt{dim}=1,\texttt{shift}=-2)\\ &+ b^{(-1)} * \texttt{cshift}(\tilde{\mathbf{E}}^c,\texttt{dim}=1,\texttt{shift}=-1)\\ &+ b^{(-0)} * \texttt{cshift}(\tilde{\mathbf{E}}^c,\texttt{dim}=1,\texttt{shift}=0)\\ &+ b^{(+1)} * \texttt{cshift}(\tilde{\mathbf{E}}^c,\texttt{dim}=1,\texttt{shift}=+1)\\ &+ b^{(+2)} * \texttt{cshift}(\tilde{\mathbf{E}}^c,\texttt{dim}=1,\texttt{shift}=+2),\end{aligned} \tag{12}$$

where `dim=1` signifies the ξ-direction, and the superscript c is used to indicate the associated variable is actually a composite variable of the three regions R, O, and I. Now if this derivative is to be evaluated for these three regions sequentially, the total number of `cshifts` for a single variable in only one direction is 3×4 (shift by 0 not counted). Considering $\tilde{\mathbf{E}}$ (or $\tilde{\mathbf{F}}$, $\tilde{\mathbf{G}}$) is a five-element vector and for a 3D problem, the derivative operator is required in a total of three directions, the `cshifts` sum up to $(3 \times 4) \times (5 \times 3)$. This implies a significant amount of communication among the processors, which is the most time consuming process on the Connection Machine. However, one can combine the $\tilde{\mathbf{E}}$'s of the three zones R, O, and I to form $\tilde{\mathbf{E}}^c$ in such a way that, when Eq. (12) is performed, the derivative $\partial\tilde{\mathbf{E}}^c/\partial\xi$ is obtained for these three zones simultaneously. In other words we do not have any idle processors; hence, they do not perform unnecessary `cshifts` for any grid point when evaluating the spatial derivatives of Eq. 4, thus cutting down the number of `cshifts` by 67%. This is all achieved by using the intrinsic `MERGE` command [9], for which the syntax is given as

$$\texttt{C} = \texttt{MERGE(A, B, mask} = \texttt{M)}, \tag{13}$$

where `A` and `B` are the source (argument) arrays of the same rank, size, and type; `M` is the logical masking array of the same rank and size as the source arrays, and `C` is the target array, which is of the same rank, size, and type as `A` and `B`. This intrinsic function basically selects the elements of the target array `C` among the elements of the source arrays `A` and `B` such that where the mask `M` is `.true.` the array `A` provides the elements of `C`, and so does the array `B` elsewhere.

Now consider the third `cshift` operation in Eq. (12), i.e., the instruction

$$\texttt{cshift}(\tilde{\mathbf{E}}^c, \texttt{dim} = 1, \texttt{shift} = 0), \tag{14}$$

which essentially gives back $\tilde{\mathbf{E}}^c$, and we desire this $\tilde{\mathbf{E}}^c$ to be $\tilde{\mathbf{E}}_{\text{int}}$ at the interior points, $\tilde{\mathbf{E}}_{\text{rad}}$ at the radiation (inflow) boundary points, and $\tilde{\mathbf{E}}_{\text{out}}$ at the outflow boundary points. Using embedded `MERGE` statements, we can write

$$\tilde{\mathbf{E}}^c = \texttt{MERGE}(\tilde{\mathbf{E}}_{\text{int}}, \texttt{MERGE}(\tilde{\mathbf{E}}_{\text{rad}}, \tilde{\mathbf{E}}_{\text{out}}, \texttt{mask} = M_{\text{rad}}), \texttt{mask} = M_{\text{int}}), \tag{15}$$

where the logical masks are defined as

$$M_{\text{rad}} = \begin{Bmatrix} \texttt{.true.} & \text{in zone R} \\ \texttt{.false.} & \text{elsewhere} \end{Bmatrix}, \qquad M_{\text{int}} = \begin{Bmatrix} \texttt{.true.} & \text{in zone I} \\ \texttt{.false.} & \text{elsewhere} \end{Bmatrix}. \tag{16}$$

According to Eq. (13), this procedure will give us exactly what we desire.

Now consider the other `cshift` operations in Eq. (12), i.e., the instruction to *shift* $\tilde{\mathbf{E}}^c$ by r positions (different from 0). We would like to have

$$\texttt{cshift}(\tilde{\mathbf{E}}^c, \texttt{dim} = 1, \texttt{shift} = r) = \begin{cases} [\tilde{\mathbf{E}}_{\text{int}}]_{i+r,j,k} & \text{at interior points} \\ [\tilde{\mathbf{E}}_{\text{rad}}]_{i+r,j,k} & \text{on inflow boundaries} \\ [\tilde{\mathbf{E}}_{\text{out}}]_{i+r,j,k} & \text{on outflow boundaries.} \end{cases} \tag{17}$$

Some care is needed to achieve the above reduction of the right-hand side expression (which requires three `cshifts`) to the left-hand side (which requires only one `cshift`) using the `MERGE` command. After all the `cshift` instructions are performed, the right-hand side terms of Eq. (12) must all be of the same type function $\tilde{\mathbf{E}}$, i.e., either $\tilde{\mathbf{E}}_{\text{rad}}$, or $\tilde{\mathbf{E}}_{\text{out}}$, or $\tilde{\mathbf{E}}_{\text{int}}$, so that the weighted sum of them gives the correct derivative for every grid point in the domain. In this case we cannot simply combine $\tilde{\mathbf{E}}$'s of the regions R, O, and I into $\tilde{\mathbf{E}}^c$ in the same manner as in Eq. (15), and then apply the `cshift` to obtain Eq. (17). Doing so for every `cshift` operation would result in an indistinguishable collection of the *shifted* $\tilde{\mathbf{E}}^c$'s on the right-hand side of Eq. (12), thus an incorrect derivative for the grid points. We, therefore, use masking arrays that are shifted a priori by r positions, but in the opposite direction to provide the `cshift` with access to the right $\tilde{\mathbf{E}}$ for every grid point across the entire computational domain.

Thus in general we define the preshifted `masks` for the three zones as

$$M_{\text{rad}}^{(r)} = \texttt{cshift}(M_{\text{rad}}, \texttt{dim} = 1, \texttt{shift} = -r) \tag{18}$$

$$M_{\text{int}}^{(r)} = \texttt{cshift}(M_{\text{int}}, \texttt{dim} = 1, \texttt{shift} = -r), \tag{19}$$

where the superscript (r) indicates the mask is to be used in *shifting* the argument $\tilde{\mathbf{E}}^c$ by r positions as required by Eq. (17), and the masks M_{rad} and M_{int} are as defined by Eq. (16).

Now the selection of the appropriate flux component or the function $\tilde{\mathbf{E}}^c$ to be shifted by r positions can be made using the `MERGE` statement:

$$\tilde{\mathbf{E}}^c = \texttt{MERGE}(\tilde{\mathbf{E}}_{\text{int}}, \texttt{MERGE}(\tilde{\mathbf{E}}_{\text{rad}}, \tilde{\mathbf{E}}_{\text{out}}, \texttt{mask} = M_{\text{rad}}^{(r)}), \texttt{mask} = M_{\text{int}}^{(r)}). \tag{20}$$

The same procedures apply for obtaining the derivatives in the other directions in parallel, with only a change of the axis (dim above) along which data is to be shifted.

The masking arrays, including the preshifted ones, are determined only once and saved throughout a run to be used in the above procedures. Although this requires extra storage for these arrays, one often trades increased memory usage for decreased CPU time.

2.5. *Artificial Dissipation*

Central differencing algorithms usually require artificial dissipation to suppress the high frequency numerical oscillations. A blend of second and fourth or second and sixth-order Jameson [20] type artificial dissipation is used in the current algorithm. The dissipation vector $\mathscr{D}$ is computed using the interior dependent variables $\mathbf{Q}_{\text{int}}$ for every grid point, including the far-field boundary points. This is because in a data parallel code, the same operations are already performed for every element of an array.

The dissipation for the far-field boundary points is then obtained from $\mathscr{D}(\mathbf{Q}_{\text{int}})$ via linearization. Thus one can write

$$\begin{aligned} \mathscr{D}(\rho') &= \mathscr{D}(\rho), \\ \mathscr{D}(u_1') &= \frac{1}{\rho_\infty}[\mathscr{D}(\rho u_1) - u_{1\infty}\mathscr{D}(\rho)], \\ \mathscr{D}(u_2') &= \frac{1}{\rho_\infty}[\mathscr{D}(\rho u_2) - u_{2\infty}\mathscr{D}(\rho)], \\ \mathscr{D}(u_3') &= \frac{1}{\rho_\infty}[\mathscr{D}(\rho u_3) - u_{3\infty}\mathscr{D}(\rho)], \\ \mathscr{D}(p') &= c_\infty^2 \mathscr{D}(\rho'), \end{aligned} \tag{21}$$

where $\mathscr{D}(\,)$ is the dissipation flux of the respective argument, and the subscript ∞ is associated with the free stream.

This procedure does not require a separate calculation of the dissipation fluxes for the far-field boundary points.

3. FAR-FIELD PREDICTIONS BY KIRCHHOFF METHOD

The Kirchhoff methods have recently found more and more use in flow induced far-field noise predictions [21].

The reason for this is twofold. First the numerical algorithms involve dissipation and dispersion errors that can accumulate as one keeps propagating waves all the way to the observer (far-field) point, which may result in inaccurate predictions. The second is the demand on computational power dictated by such a direct approach is excessive, and practical calculations are impossible. Here we avoid such problems by passing the Navier–Stokes/Euler near-field solution to a closed Kirchhoff surface to extrapolate the near-field solution to the far-field observer point assuming that the field between the two is governed by the linear wave equation. This approach requires justification of the choice of the closed Kirchhoff surface for the desired accuracy of the far-field predictions. The surface is chosen to include all the important physical effects on the acoustic waves, such as the nonlinear and nonuniform background flow effects.

An extensive review of the Kirchhoff methods used in the literature is given by Lyrintzis [22]. Here the Kirchhoff formula developed by Farassat and Myers [12] for arbitrarily moving and deforming surfaces is used and coupled with the Navier–Stokes/Euler solver of the current numerical algorithm. This formula was developed using the theory of generalized functions [23] and gives a null (zero) field inside the Kirchhoff surface.

For a nondeforming Kirchhoff surface that is in rectilinear motion, the Kirchhoff formula for the acoustic pressure p' at the observer location $\mathbf{x}$ and observer time t takes the form

$$4\pi p'(\mathbf{x}, t) = \iint_S \left[\frac{E_1}{r(1 - M_r)} + \frac{p' E_2}{r^2 (1 - M_r)} \right]_{\tau^*} dS, \quad (22)$$

where

$$r = |\mathbf{r}|, \quad \mathbf{r} = \mathbf{x} - \mathbf{y}(\tau), \quad M_r = \mathbf{M} \cdot \mathbf{r}/r, \quad (23)$$

$$E_1 = -\hat{n} \cdot \nabla p' + (\mathbf{M} \cdot \hat{n})(\mathbf{M} \cdot \nabla p') + \left[\frac{\cos\theta - \mathbf{M} \cdot \hat{n}}{c_\infty (1 - M_r)} - \frac{\mathbf{M} \cdot \hat{n}}{c_\infty} \right] \frac{\partial p'}{\partial \tau}, \quad (24)$$

$$E_2 = \frac{1 - M^2}{(1 - M_r)^2} (\cos\theta - \mathbf{M} \cdot \hat{n}) \quad (25)$$

in which $\mathbf{y}$ is the surface point coordinates, τ is the source time, $\mathbf{M}$ is the Mach number vector at which the surface travels, $\hat{n}$ is the normal vector pointing out of the control (Kirchhoff) surface, c_∞ is the free stream speed of sound, and θ is the angle between the vectors $\mathbf{r}$ and $\hat{n}$. The integrand of the above equation is to be evaluated at the control point emission time, τ^*, which is given by the root of

$$\tau - t + r(\tau)/c_\infty = 0. \quad (26)$$

The Kirchhoff surface is constructed out of the CFD grid by specifying constant ξ, η, and/or ζ surfaces. Equation (22) requires the surface parameters, such as the normal vector and surface element area. This information is directly related to the metrics given by Eq. (7). The acoustic pressure required by Eq. (22) is obtained from the flow solver, by subtracting the mean pressure from the total pressure. Equation (22) is integrated as the solution of the interior domain progresses through the R-K iterations.

In most problems the observer and the Kirchhoff surface are fixed relative to each other. Also the coordinate system is attached to the moving surface rather than the fluid. In this case the fluid passes both the observer and the Kirchhoff surface at the same speed, M_∞. Thus the Kirchhoff surface and the observer are seen to travel at $-M_\infty$, and the elapsed time for a signal to travel from a Kirchhoff surface element to the observer will always be the same in time. This greatly simplifies the numerical coupling of the flow solver and the Kirchhoff method. Thus the emission time for a Kirchhoff surface element e is given by

$$\tau_e = t - r_e/c_\infty, \quad (27)$$

where

$$r_e = \frac{-(x_1 - y_1) M_\infty + \sqrt{(x_1 - y_1)^2 + (1 - M_\infty^2) \times [(x_2 - y_2)^2 + (x_3 - y_3)^2]}}{1 - M_\infty^2}, \quad (28)$$

where x_1 is the observer distance vector component which is aligned with the free stream $\mathbf{M}_\infty$ vector and x_2 and x_3 are the transverse components. The components of the element position vector $\mathbf{y}$ are defined similarly and denoted by y_1, y_2, and y_3.

The integrations are performed assuming that the integrand in Eq. (22) is constant over a Kirchhoff surface element, and the time is discretized in the same manner as in the flow solver. Thus for the discrete acoustic pressure at the observer point and time we can write

$$p'[\mathbf{x}, n\,\Delta t] = \frac{1}{4\pi} \sum_{e=1}^{NE} P_e[(n - \mathrm{int}(R_e/c_\infty))\,\Delta t]\,\Delta S_e, \quad (29)$$

where n shows the time level, the subscript e indicates the (e)th surface element, NE is the total number elements on the discretized Kirchhoff surface, the operator int( ) rounds the real argument to the closest integer, and the summation is the discretized version of the right-hand side of Eq. (22), which corresponds to the integral. This equation requires the past time history of the integrand for each surface element, which usually implies a very large memory requirement. Here this problem is overcome by using a recursive summation approach.

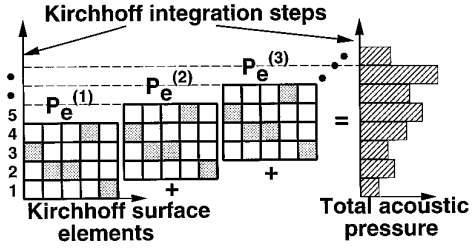

FIG. 4. Recursive Kirchhoff integration.

Each surface element e at time t of the Navier–Stokes/Euler numerical solver will contribute to the total acoustic pressure at the observer location a time period of r_e/c_∞ later. In other words, a signal of element e produced at time t will arrive at the observer at $t + r_e/c_\infty$. Thus for the discretized acoustic pressure at the observer we can write

$$p'_{\text{new}}[\mathbf{x}, (n + \text{int}(r_e/c_\infty))\,\Delta t] = p'_{\text{old}}[\mathbf{x}, (n + \text{int}(r_e/c_\infty))\,\Delta t] + \frac{1}{4\pi} P_e[n\,\Delta t]\,\Delta S_e, \tag{30}$$

where $e = 1, 2, ..., NE$, and the Kirchhoff integrations are assumed to start at $n = n_{KS}$, and end at $n = n_{KE}$, and the initial values of pressure at the observer are set to zero, i.e.,

$$p'[\mathbf{x}, (n + \text{int}(r_e/c_\infty))\,\Delta t] = 0, \quad n \le n_{KS}, e = 1, 2, ..., NE. \tag{31}$$

Then this equation will give a transient field for some time periods after $n = n_{KS}$, and before $n = n_{KE}$. This is due to the fact that we are essentially predicting future acoustic pressure contributions at the observer from each Kirchhoff surface element, which progressively sum up to the total acoustic pressure at the observer. The Kirchhoff results are then usable only in the range $n_{KS} + \Delta n_{\text{arr}} \le n \le n_{KE} - \Delta n_{\text{arr}}$, where $\Delta n_{\text{arr}} = \text{int}(r/c_\infty)_{\max} - \text{int}(r/c_\infty)_{\min}$.

The integration is schematically shown in Fig. 4, where the pressure contributions of the surface elements are shown to be scattered in a 2D matrix box, according to the number of time steps (int(r_e/c_∞), vertical axis of the box) for an element's signal (horizontal axis of the box) to arrive at the observer. The pressure contribution of each element is shown by $p_e^{(n)}$, signifying the signal of value p is emitted from the (e)th element at the nth time step. Then Eq. (30) corresponds to recursive summation of the full locations of the rows of the matrix boxes that are shifted in vertical direction (time step axis) by one step from one time step to the next.

4. RESULTS AND DISCUSSION

The capability of the numerical algorithm is demonstrated in this section. Several test cases are presented to evaluate various components of the code. The studies include an oscillating flat plate in a viscous fluid, acoustic scattering from a sphere, sound radiation from a baffled circular piston, steady state flow through an engine inlet, and sound propagation through an engine inlet.

4.1. *Oscillating Flat Plate in a Viscous Fluid*

The time accurate Navier–Stokes algorithm has been tested for Stokes' second problem, in which a flat plate in a viscous fluid oscillates in its own plane with a constant frequency. An analytical solution is available in the case of an incompressible fluid. The motion of the plate is defined as $u = U\cos(2\pi ft)$. The amplitude of the oscillation is given by $U/c_\infty = 0.3$, the Reynolds number is 10,000 and the oscillation frequency is $f = 100$ Hz. Figure 5 compares the numerical solution with the exact solution at an instant of time. The code predicted the velocity profile in the boundary layer very well. However, it has been observed from the other tests of this problem that, if not carefully modeled, the amount of artificial dissipation in the boundary layer can affect the solution significantly, taking over the physical diffusion (viscosity). Anisotropic dissipation models are suggested for viscous calculations involving solid boundaries [24].

4.2. *Scattering From a Sphere*

In this section we compare the full Euler solution of the current code with the linear analytical solution of Morris [25] for the scattering of spherical sound waves generated by a spatially distributed Gaussian source from a sphere. The source used for the Euler computations is a mass source, with its effects added to the momentum and the energy equations. The mass source is given by

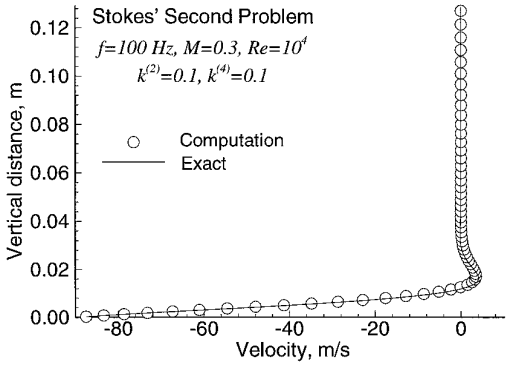

FIG. 5. Oscillating flat plate in viscous fluid: velocity profile at an instant of time, $U/c_\infty = 0.3$, Re $= 10^4$, $f = 100$ Hz.

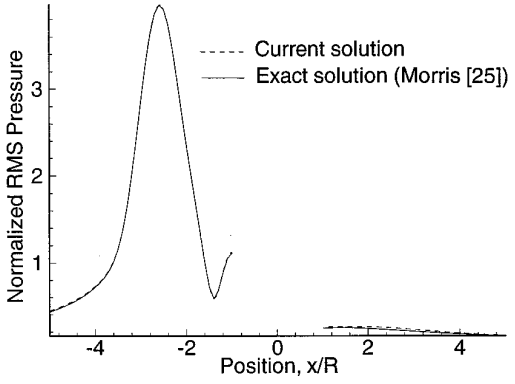

FIG. 6. Comparison of the normalized RMS of the total acoustic pressures along the x-axis (singular line of the grid).

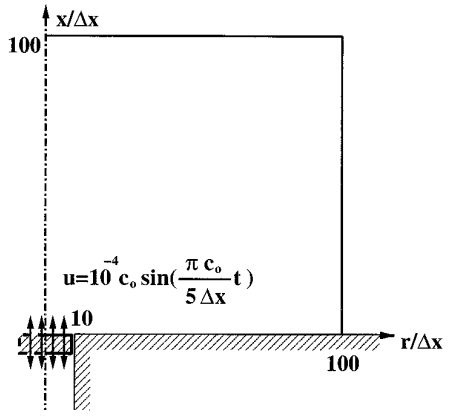

FIG. 8. Configuration of the oscillating baffled piston problem.

$$\dot{m} = \rho_\infty \exp[-4 \log 2((x + 2.56)^2 + y^2 + z^2)] \cos(2\pi f t), \tag{32}$$

with the source frequency $f = 212.6$ Hz. The sphere radius R is 1 m, and a 192×192 grid (axisymmetric) is used, with a far-field boundary at 5.5 R from the center of the sphere. Figure 6 compares the numerical and exact solutions for the root mean square of the total acoustic pressure along the x-axis (singular line) on which the center of the source is located. The sphere lies between $x = -1$ m and $x = +1$ m. The agreement between the two solutions is excellent, which is also evident from Fig. 7. This figure compares the direct Euler solution, the Kirchhoff prediction using the direct Euler solution, and the exact solution at a distance of 4.5 R from the center of the sphere. The Kirchhoff surface was chosen to be a spherical shell located at a distance of approximately $3R$ from the center. The direct CFD data starts deviating slightly toward 180° from the exact solution due to possible dissipation and dispersion errors, but the Kirchhoff prediction does not show any deviation from the exact solution, indicating the effectiveness of the hybrid method in acoustic calculations.

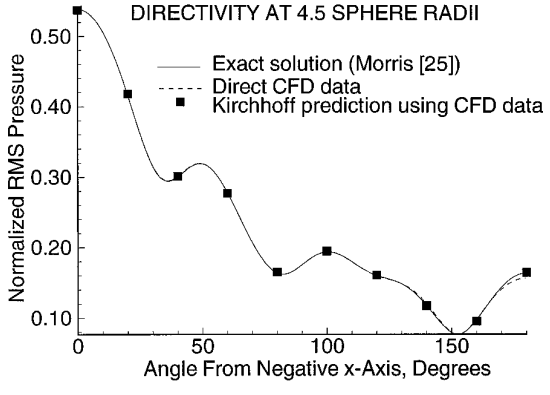

FIG. 7. Comparison of the current and exact solutions for the scattering directivity at 4.5 R distance from the sphere center.

4.3. *Oscillating Baffled Piston*

The next example is an oscillating circular piston in a wall with an axial velocity of $u = 10^{-4}\, c_\infty \sin((\pi c_\infty / 5\,\Delta x)t)$. This problem is one of the test problems of the ICASE/LaRC Workshop on Benchmark Problems in Aeroacoustics [26]. The problem is solved on a $100\,\Delta x \times 100\,\Delta x$ grid in cylindrical coordinates in the axisymmetric mode as shown in Fig. 8.

The radius of the piston is $10\,\Delta x$. The above frequency of the piston oscillation corresponds to having 10 cells per wavelength. The full Euler equations were solved together with the radiation conditions on the outer boundaries. Figure 9 compares the acoustic pressure contours of the numerical simulation with the exact solution. The pressure levels seen in the figure were scaled by the spherical distance from the center of the piston to suppress the spherical spreading effect. The far-field boundary conditions worked very well against continuous acoustic radiation from the piston. The overall agreement between the current solution and the exact is excellent.

4.4. *Engine Inlet Flow*

The ultimate goal of the current research is radiation problems from ducted fans of gas turbine engines, where one has nonuniform mean flow and rotating and propagating pressure patterns in an acoustically treated inlet. For the inlet radiation problem one has to first determine the mean (steady) flow through the inlet and then determine the unsteady field driven by the acoustic source at the fan. The acoustic field is given by the difference between the unsteady and steady fields. The code has been tested for steady flow through the General Electric engine inlet C1A at cruise and off-design conditions. A meridional section

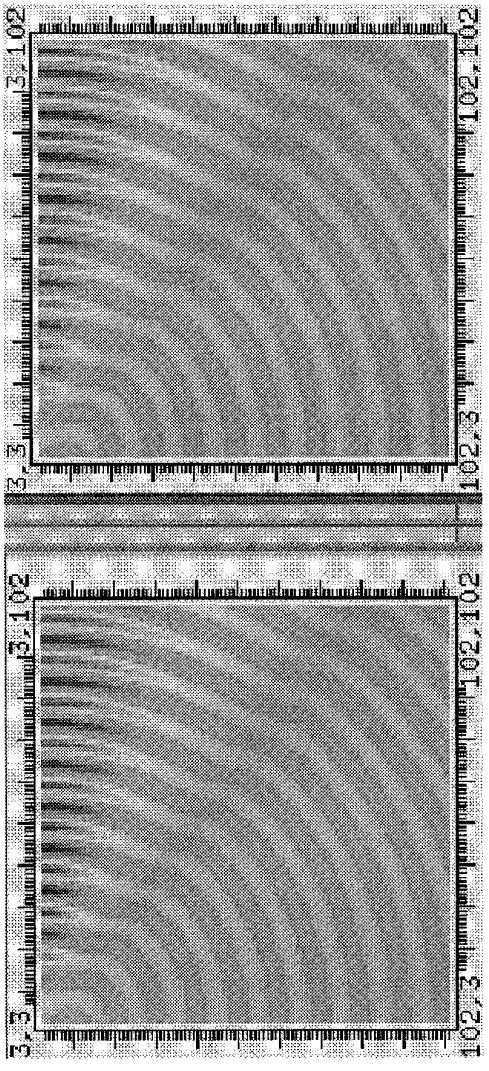

FIG. 9. Sound radiation from oscillating piston: Exact (top) and Euler (bottom) solutions.

of the C-O grid system (total of 64,000 cells) used for the calculations is shown in Fig. 10. The geometry is symmetric about the vertical plane, and the 3D grid was generated for only one-half of the inlet using an elliptic mesh generator. The Euler equations, together with nonreflecting boundary conditions, were solved using the Cartesian formulations. Local time stepping was used and symmetry boundary conditions were applied in the symmetry plane. For this problem the pressure at the fan was adjusted according to the mass flow rate that was computed at every iteration, which was started by specifying, at the fan, a constant pressure, obtained by using the 1D ideal gas dynamics relations for the given mass flow rate. The other variables were obtained by solving the interior equations. The iterations were stopped after a reasonably accurate mass flow at the fan was reached. This procedure is identical to that of Uenishi *et al.* [11].

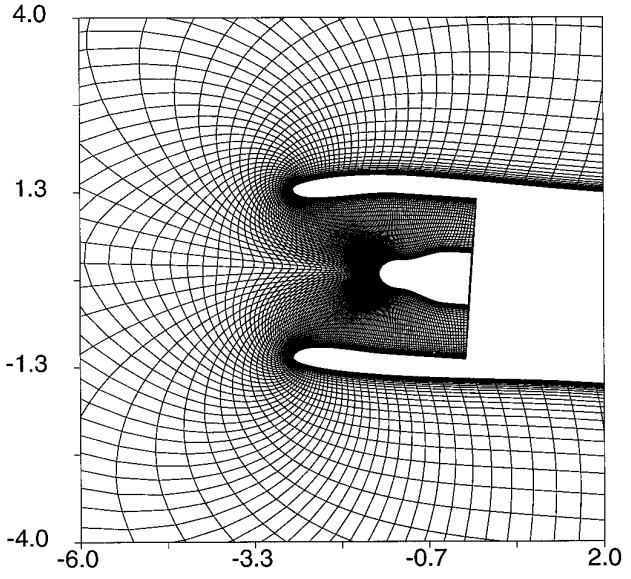

FIG. 10. A meridional view of the C-O grid about the G.E. C1A engine inlet.

Figure 11 presents the Mach number contours for the cruise condition, for which the free stream Mach number M is 0.82, the mass flow ratio MFR is 0.65, and the angle of attack α is 4.0°. The mass flow ratio is defined as the ratio of the amount of mass flux going through the inlet to the mass flux that would go through a hi-lite (inlet lip leading edge location) cross section equivalent area at the free stream conditions. Notice that the contour lines are very smooth in the vicinity of the centerline and the centerbody nose, where there exist grid singularities. This smooth solution was achieved by cancelling the mass-

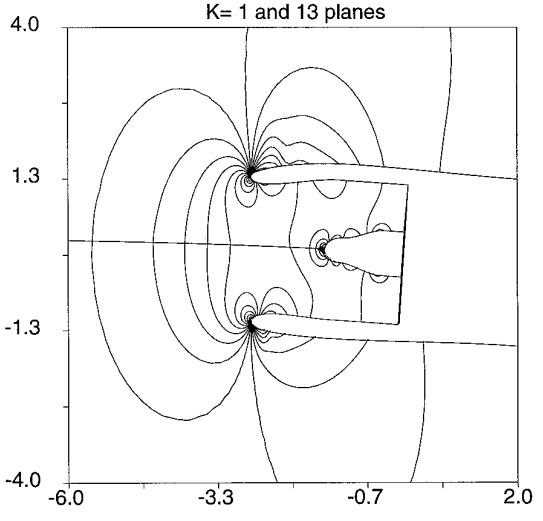

FIG. 11. Mach contours around the G.E. C1A inlet, M = 0.82, MFR = 0.65, α = 4.0°.

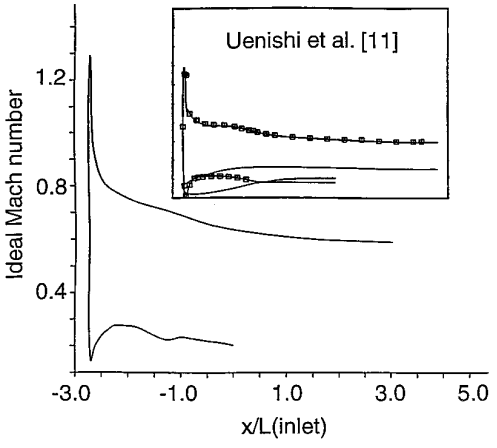

FIG. 12. Ideal Mach number distribution on the C1A cowl at the crown cut, M = 0.65, MFR = 0.351, $\alpha = 4.0°$.

source-like terms which are associated with the nonvanishing discretized divergence of the metric vectors.

Figure 12 gives the ideal Mach number distribution, which is computed using the local pressure with the 1D ideal gas dynamics relations, on the engine cowl at the crown cut for the off-design condition, M = 0.65, MFR = 0.351, $\alpha = 4.0°$. This condition and the above correspond to those of Uenishi *et al.* [11]. Qualitative agreement for the off-design conditions between their result and the current result is very good as shown in Fig. 12.

4.5. *Propagation Through Inlet*

In this section we demonstrate an example sound propagation problem through an axisymmetric inlet, JT15D [27]. This example includes the calculation of a steady state flow field at M = 0.204 first and then applying at the fan an acoustic source [6] of the combined (0, 0) and (0, 1) modes with a 180° phase difference, each with a 300 N/m^2 amplitude and at a blade passing frequency (BPF) of 5 kHz. The mass flow rate for this case is 15 kg/s. The grid ($384 \times 96 \times 1$) used for the calculations is shown in Fig. 13, which is relatively coarse near the outer boundaries. This grid was generated using conformal mapping [28, 16].

The Euler equations and nonreflecting boundary conditions in cylindrical coordinates were used for the calculations presented in this section. The steady state solution was carried out using a multigrid convergence acceleration procedure that retains the fourth-order spatial accuracy [16]. Also the 1D characteristic boundary conditions of Giles [29] were used at the fan to achieve an accurate mass flow through the inlet, as well as to prevent possible reflections from the fan-face. The corresponding steady state pressure contours are shown in Fig. 14.

Once the steady state was reached, the source was turned on and the calculations were continued. The acoustic pressure contours, illustrated in Fig. 15, were obtained by subtracting the steady state pressure from the pressure of the time accurate calculations. It is evident from both Figs. 14 and 15 that the steady and the acoustic pressure contours are extremely smooth in the vicinity of the centerline, showing that the centerline treatment of the current algorithm is effective.

As the waves travel upstream cancelations and reinforcements occur due to phase changes. This can be seen in detail in the centerline pressure plot (Fig. 16), where the steady state and a snapshot of the time accurate pressures are shown together.

As well as the spherical spreading effects, large cells near the upstream far-field boundaries play a role in the diminish-

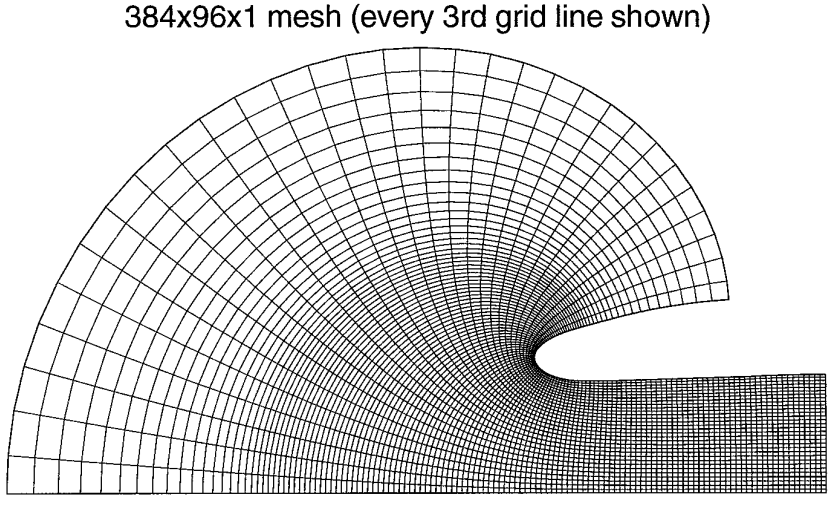

FIG. 13. Grid around the JT15D inlet.

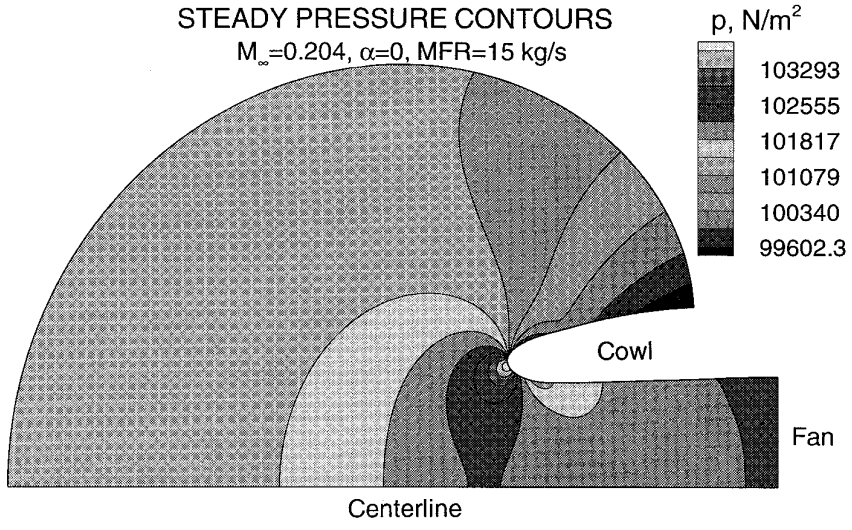

FIG. 14. Steady state pressure contours around the JT15D inlet, $M_\infty = 0.204$, $\alpha = 0°$, Mass Flow Rate = 15 kg/s.

ing of the pressure amplitude, since large cells cannot support the higher wave frequencies. This is in fact the main motivation for using a far-field extrapolation technique, such as the Kirchhoff method, taking the near-field solution, to accurately predict the far-field noise. Further validation studies involving engine inlet noise calculations using the Kirchhoff method are reported elsewhere [6].

4.6. *Performance*

The performance characteristics of the current code are documented in Table I for various segments of the code. On a time-shared 256-node CM-5 the code uses about a total of 3.26 μs per grid point per time step when the Euler equations are integrated together with nonreflecting boundary conditions on the far-field boundaries and the Kirchhoff calculations are performed to extrapolate the near-field solution to the far-field. The computational mesh used in the timing of the code was only one of 13 periodic domains in the circumferential direction, and one domain involved $256 \times 32 \times 16$ grid points, from which two Kirchhoff surfaces were constructed. Since Kirchhoff integrations are performed on a closed surface, each Kirchhoff surface was extended to cover these 13 periodic domains,

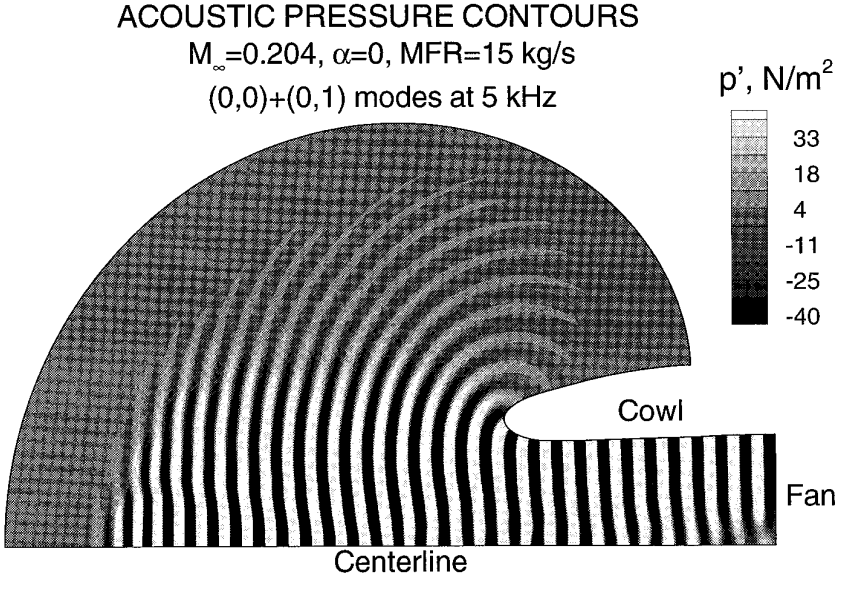

FIG. 15. Propagation of (0, 0) + (0, 1) modes (plane waves) through the JT15D inlet. BPF = 5 kHz, $M_\infty = 0.204$, $\alpha = 0°$, Mass Flow Rate = 15 kg/s.

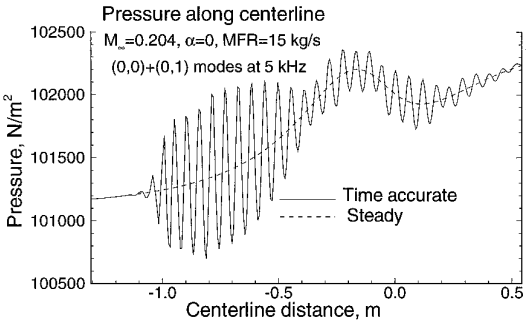

FIG. 16. The steady and the time accurate pressures along the centerline of the JT15D inlet. BPF = 5 kHz, $M_\infty = 0.204$, $\alpha = 0°$, Mass Flow Rate = 15 kg/s.

then involving a total of $13 \times (32 \times 16)$ surface elements to integrate the acoustic field on them to 12 far-field observer points. The total memory usage was 450 M Bytes.

The majority of the CPU time was spent on calculating the residuals of the governing equations (25.2%) and on the Kirchhoff calculations (25.4%). The optimization techniques described in Section 2.4 provided approximately 60% reductions in computing the residuals of the governing equations, and the procedure described in Section 2.5 approximately 50% in computing the artificial dissipation.

5. CONCLUSIONS

An efficient, hybrid, computational aeroacoustics algorithm has been described. It solves the 3D Navier–Stokes/Euler equations on a body-fitted coordinate system using temporally and spatially fourth-order accurate, Runge–Kutta time integration and cell-centered finite difference schemes. The algorithm has been developed for aeroacoustics applications on parallel processors and coupled with a Kirchhoff method for far-field predictions. Several optimization techniques have been introduced in this development, which yielded a very efficient algorithm. The code uses about 2.43 μs per grid point per time step on a 256-node time-shared CM-5 for Euler calculations and about 3.26 μs and 450 M Bytes of memory for Euler calculations with Kirchhoff integrations. The use of the `MERGE` command has provided significant CPU savings in computing the residuals of the governing equations. The use of this intrinsic statement is suggested in similar CFD algorithms for efficient coding in HPF.

It has been found that, on meshes having singular lines, such as the centerline of an engine inlet, formulating the governing equations in cylindrical coordinates and using cell-centered finite differencing permits proper treatment of the grid singularity. This is extremely crucial for successful aeroacoustic simulations of the turbofan engine inlets.

It has been demonstrated with various results that the algorithms are capable of calculating both steady and unsteady (acoustics) problems.

TABLE I

Breakdown of CPU Time Spent on Various Segments of the Code for a $256 \times 32 \times 16$ Grid with 13 Periodic Kirchhoff Surfaces (256 Node CM-5 in Time-Sharing Mode)

Work	CPU, μ s/step − cell	Percentage
Inv. fluxes	0.82	25.2
Dissipation	0.17	5.2
Wall BCs	0.46	14.1
Ghost cells	0.53	16.3
Kirchhoff	0.83	25.4
Other	0.45	13.8
Total	3.26	100.0

APPENDIX

In this section the terms $\mathbf{E}$, $\mathbf{F}$, $\mathbf{G}$, $\mathbf{S}_B$, and $\mathbf{S}_C$ of the far-field nonreflecting boundary conditions, Eq. (1), are given with a choice of the Cartesian or the cylindrical coordinate systems for arbitrary 3D subsonic flow problems, with the angle of attack α, being measured as the angle between the free stream vector and its projection on the x–z plane in the Cartesian system, and the yaw angle ψ, being measured as the angle between the x-axis and the projection of the free stream vector on the x–z plane. Any quantity with the subscript ∞ indicates the undisturbed value (free stream). Any dependent perturbation variable q' is the difference between the total q and the undisturbed q, i.e., $q' = q - q_\infty$:

For Cartesian coordinates,

$$m = 0, \quad (x_1, x_2, x_3) = (x, y, z),$$
$$(u_1, u_2, u_3) = (u, v, w);$$

For cylindrical coordinates,

$$m = 1, \quad (x_1, x_2, x_3) = (x, r, \theta),$$
$$(u_1, u_2, u_3) = (u, v_r, w_\theta).$$

Radiation Boundary Conditions

$$\mathbf{Q}_{\text{rad}} = [\rho', u_1', u_2', u_3', p']^{\text{T}},$$
$$\mathbf{E}_{\text{rad}} = \hat{c}_1 \mathbf{Q}_{\text{rad}}, \quad \mathbf{F}_{\text{rad}} = \hat{c}_2 \mathbf{Q}_{\text{rad}},$$
$$\mathbf{G}_{\text{rad}} = \hat{c}_3 \mathbf{Q}_{\text{rad}}, \quad \mathbf{S}_{B,\text{rad}} = \hat{c}_4 \mathbf{Q}_{\text{rad}},$$

$$\mathbf{S}_{C,\text{rad}} = \begin{Bmatrix} \hat{c}_2\rho' \\ \hat{c}_2 u_1' \\ \hat{c}_2 u_2' - \hat{c}_3 u_3' \\ \hat{c}_2 u_3' + \hat{c}_3 u_2' \\ \hat{c}_2 p' \end{Bmatrix},$$

where

$$\hat{c}_1 = \tilde{c}_1,$$
$$\hat{c}_2 = \tilde{c}_2 \cos mx_3 + \tilde{c}_3 \sin mx_3,$$
$$\hat{c}_3 = \tilde{c}_3 \cos mx_3 - \tilde{c}_2 \sin mx_3,$$
$$\hat{c}_4 = x_2^m \tilde{c}_4 - x_2^m \left[\frac{\partial \hat{c}_1}{\partial x_1} + \frac{\partial \hat{c}_2}{\partial x_2} + \frac{1}{x_2^m}\frac{\partial \hat{c}_3}{\partial x_3} + \frac{m}{x_2^m}\hat{c}_2 \right],$$

in which

$$\tilde{c}_4 = c_4',$$

$$\begin{Bmatrix} \tilde{c}_1 \\ \tilde{c}_2 \\ \tilde{c}_3 \end{Bmatrix} = \Omega \begin{Bmatrix} c_1' \\ c_2' \\ c_3' \end{Bmatrix},$$

with

$$\Omega = \begin{bmatrix} \cos\alpha\cos\psi & -\sin\alpha\cos\psi & -\sin\psi \\ \sin\alpha & \cos\alpha & 0 \\ \cos\alpha\sin\psi & -\sin\alpha\sin\psi & \cos\psi \end{bmatrix},$$

$$c_1' = \frac{x'}{d'A'}, \quad c_2' = \frac{y'}{d'A'}, \quad c_3' = \frac{z'}{d'A'},$$

$$c_4' = \frac{1}{d'A'}, \quad d'^2 = \frac{x'^2}{1 - M_\infty^2} + y'^2 + z'^2,$$

$$A' = \frac{1}{c_\infty \sqrt{1 - M_\infty^2}} \left(1 - \frac{x'}{d'} \frac{M_\infty}{\sqrt{1 - M_\infty^2}} \right),$$

$$\begin{Bmatrix} x' \\ y' \\ z' \end{Bmatrix} = \Omega^{-1} \begin{Bmatrix} x - x_{\text{ref}} \\ y - y_{\text{ref}} \\ z - z_{\text{ref}} \end{Bmatrix},$$

α = angle of attack,

ψ = yaw angle,

x, y, z = Cartesian coordinates,

$(x, y, z)_{\text{ref}}$ = reference point (Approximate Source Location).

Outflow Boundary Conditions

$$\mathbf{Q}_{\text{out}} = \mathbf{Q}_{\text{rad}}$$

$$\mathbf{E}_{\text{out}} = \begin{Bmatrix} u_{\infty 1}(\rho' - p'/c_\infty^2) \\ u_{\infty 1} u_1' + p'/\rho_\infty \\ u_{\infty 1} u_2' \\ u_{\infty 1} u_3' \\ \hat{c}_1 p' \end{Bmatrix},$$

$$\mathbf{F}_{\text{out}} = \begin{Bmatrix} u_{\infty 2}(\rho' - p'/c_\infty^2) \\ u_{\infty 2} u_1' \\ u_{\infty 2} u_2' + p'/\rho_\infty \\ u_{\infty 2} u_3' \\ \hat{c}_2 p' \end{Bmatrix}$$

$$\mathbf{G}_{\text{out}} = \begin{Bmatrix} u_{\infty 3}(\rho' - p'/c_\infty^2) \\ u_{\infty 3} u_1') \\ u_{\infty 3} u_\infty' \\ u_{\infty 3} u_3' + p'/\rho_\infty \\ \hat{c}_3 p' \end{Bmatrix},$$

$$\mathbf{S}_{B,\text{out}} = \begin{Bmatrix} -\frac{x_2^m}{c_\infty^2}\frac{\partial p'}{\partial t} \\ 0 \\ 0 \\ 0 \\ \hat{c}_4 p' \end{Bmatrix},$$

$$\mathbf{S}_{C,\text{out}} = \begin{Bmatrix} u_{\infty 2}(\rho' - p'/c_\infty^2) \\ u_{\infty 2} u_1' \\ u_{\infty 2} u_2' - u_{\infty 3} u_3' \\ u_{\infty 2} u_3' + u_{\infty 3} u_2' \\ \hat{c}_2 p' \end{Bmatrix},$$

where

$$u_{\infty 1} = u_\infty,$$
$$u_{\infty 2} = v_\infty \cos mx_3 + w_\infty \sin mx_3,$$
$$u_{\infty 3} = w_\infty \cos mx_3 - v_\infty \sin mx_3,$$
$$u_\infty = |\mathbf{V}_\infty| \cos\alpha \cos\psi,$$
$$v_\infty = |\mathbf{V}\infty| \sin\alpha,$$
$$w_\infty = |\mathbf{V}_\infty| \cos\alpha \sin\psi,$$
$$|\mathbf{V}_\infty| = \mathrm{M}_\infty c_\infty,$$
$$\mathrm{M} = \text{Mach number}, \quad c = \text{speed of sound}.$$

ACKNOWLEDGMENTS

The authors gratefully acknowledge support from NASA Langley Research Center Grant Number NAG-1-1367. Gratitude is also extended to K. Uenishi of General Electric Aircraft Engines, Cincinnati, OH, for providing the C1A engine inlet surface geometry. The use of the computational resources of the Numerical Aerodynamics Simulation Program at the NASA Ames Research Center and the National Center for Supercomputing Applications at the University of Illinois at Urbana-Champaign is also acknowledged.

REFERENCES

1. D. K. McLaughlin, G. L. Morrison, and T. R. Troutt, *J. Fluid Mech.* **69,** 73 (1975).
2. E. Gutmark, K. C. Schadow, and C. J. Bicek, *AIAA J.* **28**(7), 1163 (1990).
3. R. R. Mankbadi, M. E. Hayer, and L. A. Povinelli, *AIAA J.* **32**(5), 897 (1994).
4. J. M. Tyler and T. G. Sofrin, *SAE Trans.* **70,** 309 (1962).
5. W. Eversman, A. V. Parret, J. S. Preisser, and R. J. Silcox, *Trans. ASME,* **107,** 216 (1985).
6. Y. Özyörük and L. N. Long, CEAS/AIAA Paper 95-063, 1995 *AIAA J.* (1996), to appear.
7. A. Bayliss and E. Turkel, *J. Comput. Phys.* **48,** 182 (1982).
8. C. K. W. Tam and J. C. Webb, *J. Comput. Phys.* **107,** 262 (1993).
9. *CM Fortran Language Reference Manual, Version 2.1,* (Thinking Machines Corporation, Cambridge, MA, January 1994).
10. C. H. Koelbel, D. B. Loveman, and R. S. Schreiber, *The High Performance Fortran Handbook* (MIT Press, Cambridge, MA, 1994).
11. K. Uenishi, M. S. Pearson, T. R. Lehnig, and R. M. Leon, *J. Propulsion Power* **8,** 175 (1992).
12. F. Farassat and M. K. Myers, *J. Sound Vibration* **123,** 451 (1988).
13. M. Vinokur, *J. Comput. Phys.* **14,** 105 (1974).
14. F. G. Hu, M. Y. Hussaini, and J. Manthey, NASA Contractor Report 195022, 1994 (unpublished).
15. Y. Özyörük and L. N. Long, "Computational Aeroacoustics on Massively Parallel Computers," in *ICASE/LaRC Workshop on Benchmark Problems in Computational Aeroacoustics (CAA)*, edited by J. C. Hardin, J. R. Ristorcelli, and C. K. W. Tam, NASA CP-3300, 1995, p. 261.
16. Y. Özyörük, Ph.D. thesis, Pennsylvania State University, 1995 (unpublished).
17. Y. Özyörük and L. N. Long, AIAA Paper 94-0462, 1994 (unpublished).
18. J. Myczkowski, M. Bromley, and D. McCowan, AIAA Paper 91-0436, 1991 (unpublished).
19. T. S. Chyczewski and L. N. Long, CEAS/AIAA Paper 94-2265, 1994 (unpublished).
20. A. Jameson, W. Schmidt, and E. Turkel, AIAA Paper 81-1259, 1981 (unpublished).
21. Y. Xue and A. S. Lyrintzis, *AIAA J.* **32,** 1350 (1994).
22. A. S. Lyrintzis, *J. Fluids Eng.* **116,** 665 (1994).
23. F. Farassat, Technical Report NASA TP-3428, NASA, May 1994 (unpublished).
24. L. Martinelli, Ph.D. thesis, Princeton University, 1987 (unpublished).
25. P. J. Morris, *J. Acoust. Soc. Am.* **98**(6) (1995).
26. J. C. Hardin, J. R. Ristorcelli, and C. K. W. Tam (eds.), *ICASE/LaRC Workshop on Benchmark Problems in Computational Aeroacoustics (CAA),* NASA CP-3300, 1995.
27. J. S. Preisser, R. J. Silcox, W. Eversman, and A. V. Parret, AIAA Paper 84-0499, 1984 (unpublished).
28. D. C. Ives and W. A. Menor, AIAA Paper 81-0097, 1981 (unpublished).
29. M. B. Giles, *AIAA J.* **28**(12), 2050 (1990).