JOURNAL OF AIRCRAFT
Vol. 33, No. 1, January–February 1996

Data-Parallel Total Variation Diminishing Method for Sonic Boom Calculations

Anthony R. Pilon*
University of Minnesota, Minneapolis, Minnesota 55455
and
Anastasios S. Lyrintzis†
Purdue University, West Lafayette, Indiana 47907

Sonic boom predictions are shown for the near and midfield and comparisons are made with experimental data. The computations are performed on the Thinking Machines' CM-5 massively parallel supercomputer to utilize its large available memory and high floating point performance. A second-order-accurate total variation diminishing scheme is used to solve the Euler equations in the computations. Additionally, a recently developed implicit method, based on the LU–SGS algorithm, is used to speed the convergence and accuracy of the steady-state computations. The method is shown to work well on near- and midfield sonic boom predictions for several test cases.

Introduction

THE projected use of the high speed civil transport (HSCT) has drawn attention to the problems associated with the noise due to sonic booms.[1] An accurate and efficient sonic boom prediction methodology is needed for the assessment of various proposed HSCT configurations. One such method that utilizes the power of massively parallel supercomputers is presented here.

A review of current sonic boom prediction methods is given by Plotkin.[2] Many of these methods are based on the modified linear analysis of Whitham[3] and Walkden.[4] Other methods have been developed based on a modified method of characteristics that approximately account for the effects of three-dimensional flows. Experimental and analytical studies have shown that these methods lose effectiveness as freestream Mach number approaches 3. In high Mach number regions there are strong shock waves with significant entropy generation. The linear-based methods neglect these production terms.[5] A modified method of characteristics has been developed that can account for these terms,[6] but it requires nonlinear near-field initial data that is difficult to obtain computationally or experimentally.

Several authors have developed prediction methods based on near-field solutions of the Euler or Navier–Stokes equations.[7,8] Most of these methods involve marching in one spatial dimension with an implicit Euler scheme or a parabolized Navier–Stokes (PNS) algorithm. These solutions are used in conjuction with Whitham's F-function to evaluate the far-field pressure signature. The existing prediction methods are unattractive for several reasons. First, additional accuracy in the marching direction greatly increases the computational time. Also, they do not employ methods designed to resolve shock waves effectively. Finally, they are not easily implemented on massively parallel supercomputers.

Accurate shock resolution is needed in the computational fluid dynamic (CFD) methods to determine the far-field pressure signature effectively. Since the prediction methods are dependent on shock resolution it is desirable to use a computational method that is designed to capture shock waves. The total variation diminishing (TVD) schemes developed by Harten,[9] Yee,[10] and others are well suited for this purpose. In this study a second-order-accurate TVD scheme is used to determine the pressure field about several models. These solutions are then compared with experiment.

Also important in accurate prediction is the discretization of the flow domain. It would be beneficial to have as large a number of computational cells as possible while performing the calculations as quickly as possible. For this reason the computations have been performed on a massively parallel supercomputer with a very large available memory and high-peak floating point performance.

The flowfields in this study are steady. Because of this, an implicit temporal differencing method is employed. Such a method greatly reduces the time to compute a steady solution. One implicit method, specifically designed for use on massively parallel computers, is utilized here. The implicit method does not effect spatial differencing, and so it works well with the TVD scheme.

In this article the benefits of using a TVD scheme in conjunction with an implicit method on massively parallel supercomputers will be shown. It is possible to obtain accurate flow predictions at a small computational cost. In this article results are presented for three-dimensional calculations of axisymmetric flowfields. A full three-dimensional flowfield was presented in an earlier version of this article.[11]

TVD Scheme

The Harten–Yee modified-flux scheme used herein is outlined here. For simplicity, a two-dimensional scheme is derived. The extension to three dimensions is straightforward.

Consider the two-dimensional Euler equations

$$\frac{\partial U}{\partial t} + \frac{\partial F}{\partial x} + \frac{\partial G}{\partial y} = 0$$

Presented as Paper 95-0833 at the AIAA 33rd Aerospace Sciences Meeting and Exhibit, Reno, NV, Jan. 9–12, 1995; received Jan. 23, 1995; revision received July 13, 1995; accepted for publication July 14, 1995. Copyright © 1995 by A. R. Pilon and A. S. Lyrintzis. Published by the American Institute of Aeronautics and Astronautics, Inc., with permission.

*Graduate Research Assistant, Department of Aerospace Engineering and Mechanics. Student Member AIAA.

†Associate Professor, School of Aeronautics and Astronautics. Senior Member AIAA.

$\boldsymbol{U}$, $\boldsymbol{F}$, and $\boldsymbol{G}$ are the vectors of conserved quantities, and x and y direction fluxes. Thus,

$$\boldsymbol{U} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ E \end{bmatrix} \quad \boldsymbol{F} = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ (E + p)u \end{bmatrix} \quad \boldsymbol{G} = \begin{bmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ (E + p)v \end{bmatrix}$$

Where ρ is the density, u and v the x and y velocities, p the pressure, and E total energy. A first-order, explicit, finite volume formulation is then

$$\Delta \boldsymbol{U}_{i,j}/\Delta t = -(1/V_{i,j})(\boldsymbol{F}_{i+1/2,j}S_{i+1/2,j} - \boldsymbol{F}_{i-1/2,j}S_{i-1/2,j} + \boldsymbol{G}_{i,j+1/2}S_{i,j+1/2} - \boldsymbol{G}_{i,j-1/2}S_{i,j-1/2}) \tag{1}$$

Where $V_{i,j}$ is the volume of the i, j cell, S is the interface area, and Δt is the time step.

A second-order-accurate scheme is produced with modifications to the fluxes. $\tilde{\boldsymbol{F}}$ and $\tilde{\boldsymbol{G}}$ are now

$$\tilde{\boldsymbol{F}}_{i+1/2,j} = \tfrac{1}{2}(\boldsymbol{F}_{i+1,j} + \boldsymbol{F}_{i,j} + \boldsymbol{R}_{i+1/2,j}\boldsymbol{\Phi}_{i+1/2,j})$$

$$\tilde{\boldsymbol{G}}_{i,j+1/2} = \tfrac{1}{2}(\boldsymbol{G}_{i,j+1} + \boldsymbol{G}_{i,j} + \boldsymbol{R}_{i,j+1/2}\boldsymbol{\Phi}_{i,j+1/2})$$

Here, $\boldsymbol{R}$ is the right eigenvector matrix of the flux Jacobian evaluated at the cell interface. Roe's symmetric averaging[12] is used to determine the flow properties at the cell interface.

Conventional high-order-accurate schemes require the addition of linear dissipation, or artificial viscosity to prevent instabilities when calculating flow discontinuities. In the TVD scheme the necessary dissipation is added only where it is needed, near discontinuities. This dissipation enters the calculations through the vector $\boldsymbol{\Phi}$. Terms in $\boldsymbol{\Phi}$ have the effect of limiting the influence of certain cells in the computational stencil. These terms are the limiter functions. $\boldsymbol{\Phi}$ is defined by

$$\phi^l_{i+1/2,j} = \sigma(\lambda^l_{i+1/2,j})(g^l_{i+1,j} = g^l_{i,j}) - \psi(\lambda^l_{i+1/2,j} + \gamma^l_{i+1/2,j})\alpha^l_{i+1/2,j} \tag{2}$$

where

$$\psi(z) = \begin{cases} |z| & |z| \geq \delta_1 \\ (z^2 + \delta_1^2)/2\delta_1 & |z| < \delta_1 \end{cases} \tag{3}$$

Here, the index l is 1–4, corresponding to the components of the vector of conserved quantities. Also, λ^l are the eigenvalues of the flux Jacobian. α and γ are defined by

$$\alpha_{i+1/2,j} = \boldsymbol{R}^{-1}_{i+1/2,j}(\boldsymbol{U}_{i+1,j} - U_{i,j}) \tag{4}$$

$$\gamma^l_{i+1/2,j} = \frac{\sigma(\lambda^l_{i+1/2,j})(g^l_{i+1,j} - g^l_{i,j})}{\alpha^l_{i+1/2,j}} \tag{5}$$

When $\alpha^l = 0$, $\gamma = 0$. Also,

$$\sigma(z) = \begin{cases} \frac{1}{2}[\psi(z) - (\Delta t/\Delta x)z^2] & \text{Explicit} \\ \frac{1}{2}\psi(z) & \text{Implicit} \end{cases} \tag{6}$$

In Eq. (6) the first definition of σ produces second-order-accuracy in time as well as space. While this would seem to be an unnecessary calculation when using an implicit method, it does improve convergence properties. Thus, the first definition is used throughout this study. In Eq. (3) the term δ_1 is an entropy correction parameter. This prevents entropy violations in steady and nearly steady shocks. Values of 0.10–0.125 were used in this study, as recommended by Yee.[10]

The terms g^l in Eq. (2) have not yet been addressed. These are the limiter functions. Near discontinuities, the limiter functions reduce or eliminate the effects of points in the computational stencil that fall on the opposite side of a discontinuity. Many forms of limiter function are available. The easiest and most robust of these is

$$g^l_{i,j} = \text{minmod}(\alpha^l_{i-1/2,j}, \alpha^l_{i+1/2,j}) \tag{7}$$

$$\text{minmod}(x, y) = \text{sgn}(x) \cdot \max\{0, \min[|x|, y\ \text{sgn}(x)]\}$$

Many other limiter functions are available,[10] but they all serve the same purpose, and most are not as stable as Eq. (7); therefore, this is the only limiter used in this study. It should be noted that although the limiter functions preserve computational stability, they also cause the scheme to lose second-order-accuracy in the region of discontinuities. Thus, TVD schemes are not globally high-order-accurate.

Massively Parallel Supercomputer

Before discussion of the implicit algorithm it is necessary to discuss the Thinking Machines' CM-5 massively parallel supercomputer used in the calculations. This computer, located at the Army High Performance Computing Research Center, is able to compute in single instruction, multiple data (SIMD) mode, or in multiple instruction, multiple data (MIMD) mode. It is composed of 896 32-Mflop processors, arranged in a fat tree architecture. Each processor has four vector units, which leads to a maximum of 128 Mflop per processor. Thus, the 512 processor partition (the largest available) has a peak theoretical floating point performance of 65.5 Gflop. Although only fractions of this value are obtained for practical problems, the CM-5 does present the researcher with a large amount of available memory (16 Gbyte) and computational speed. It is necessary to keep interprocessor communication to a minimum with this architecture, as there is a large latency and the communication bandwidth is only 5 Mbyte/s.

The high theoretical operating speed and large available memory make the CM-5 an appealing platform for CFD problems. However, it is difficult to implement an implicit method on a massively parallel computer efficiently. This is because true implicit methods involve the solution of very large, sparse matrices. Solution of these matrices will usually leave many processors idle and will greatly diminish performance. It is possible to decrease the idle time by decomposing the computational domain into subdomains and then solving the implicit problem in each subdomain. This requires a MIMD algorithm. Additionally, there will be a great deal of necessary interprocessor communication that will also hurt performance.

An easier solution to the communication difficulty can be found in the SIMD mode. It is possible to develop a data-parallel (DP) implicit method that can operate in this mode. The method uses only nearest-neighbor communications, which greatly reduces the overall communication time of the code. The method can be programmed in a high-level language like Fortran90. This allows one to utilize optimized communication commands like CSHIFT. There is one drawback to the DP approach. To obtain optimal performance a structured grid must be used to maintain the nearest-neighbor communications. The DP implicit algorithm is outlined next.

Data-Parallel Lower–Upper Relaxation Method

As mentioned previously, true implicit methods require the solution of a very large, sparse matrix. To eliminate the interprocessor communication inherent in such a solution an approximate implicit method has been developed for DP supercomputers. This new method, developed by Candler et al.[13] is based on the lower–upper symmetric Gauss–Seidel (LUSGS) method of Yoon and Jameson.[14] In the LUSGS method, the flux Jacobians are approximated to eliminate costly matrix inversions.

Consider a fully implicit representation of the Euler equations

$$\frac{U^{n+1} - U^n}{\Delta t} + \frac{\partial F^{n+1}}{\partial x} + \frac{\partial G^{n+1}}{\partial y} = 0$$

The implicit flux vectors are now linearized by

$$\boldsymbol{F}^{n+1} \simeq \boldsymbol{F}^n + \left(\frac{\partial \boldsymbol{F}}{\partial \boldsymbol{U}}\right)^n (\boldsymbol{U}^{n+1} - \boldsymbol{U}^n) = \boldsymbol{F}^n + \boldsymbol{A}^n \delta \boldsymbol{U}^n$$

$$\boldsymbol{G}^{n+1} \simeq \boldsymbol{G}^n + \left(\frac{\partial \boldsymbol{G}}{\partial \boldsymbol{U}}\right)^n (\boldsymbol{U}^{n+1} - \boldsymbol{U}^n) = \boldsymbol{G}^n + \boldsymbol{B}^n \delta \boldsymbol{U}^n$$

These flux vectors are then split with a modified Steger–Warming approach[15]:

$$\boldsymbol{F} = \boldsymbol{F}_+ + \boldsymbol{F}_- \qquad \boldsymbol{A} = \boldsymbol{A}_+ + \boldsymbol{A}_-$$

Thus, the implicit finite volume formulation is

$$\begin{aligned} &\delta \boldsymbol{U}^n_{i,j} + (\Delta t/V_{i,j})\{(\boldsymbol{A}_{+i+1/2,j} S_{i+1/2,j} \delta \boldsymbol{U}_{i,j} \\ &\quad - \boldsymbol{A}_{+i-1/2,j} S_{i-1/2,j} \delta \boldsymbol{U}_{i-1,j}) + (\boldsymbol{A}_{-i+1/2,j} \boldsymbol{S}_{i+1/2,j} \delta \boldsymbol{U}_{i+1,j} \\ &\quad - \boldsymbol{A}_{-i-1/2,j} S_{i-1/2,j} \delta \boldsymbol{U}_{i,j}) + (\boldsymbol{B}_{+i,j+1/2} S_{i,j+1/2} \delta \boldsymbol{U}_{i,j} \\ &\quad - \boldsymbol{B}_{+i,j-1/2} S_{i,j-1/2} \delta \boldsymbol{U}_{i,j-1}) + (\boldsymbol{B}_{-i,j+1/2} S_{i,j+1/2} \delta \boldsymbol{U}_{i,j+1} \\ &\quad - \boldsymbol{B}_{-i,j-1/2} S_{i,j-1/2} \delta \boldsymbol{U}_{i,j})\}^n = \Delta U \end{aligned} \tag{8}$$

The Jacobians appearing on the left side of Eq. (8) can now be approximated. These approximations are

$$\boldsymbol{A}_+ = \tfrac{1}{2}(\boldsymbol{A} + \rho_A \boldsymbol{I}) \qquad \boldsymbol{A}_- = \tfrac{1}{2}(\boldsymbol{A} - \rho_A \boldsymbol{I})$$

where ρ_A is the spectral radius of $\boldsymbol{A}$, $|u| + a$, where a is the local speed of sound and $\boldsymbol{I}$ is the identity matrix. Then the differences between Jacobians become, e.g.,

$$\boldsymbol{A}_+ - \boldsymbol{A}_- = \rho_A \boldsymbol{I} = (|u| + a)\boldsymbol{I}$$

Moving the remaining approximated Jacobians to the right side produces

$$\begin{aligned} &(\boldsymbol{I} + \lambda_A \boldsymbol{I} + \lambda_B \boldsymbol{I})^n_{i,j} \delta \boldsymbol{U}^n_{i,j} = \Delta \boldsymbol{U}^n_{i,j} \\ &\quad + (\Delta t/V_{i,j})(\boldsymbol{A}^n_{+i-1/2,j} S_{i-1/2,j} \delta \boldsymbol{U}^n_{i-1,j} \\ &\quad - \boldsymbol{A}^n_{-i+1/2,j} S_{i+1/2,j} \delta \boldsymbol{U}^n_{i+1,j} + \boldsymbol{B}^n_{+i,j-1/2} S_{i,j-1/2} \delta \boldsymbol{U}^n_{i,j-1} \\ &\quad - \boldsymbol{B}^n_{-i,j+1/2} S_{i,j+1/2} \delta \boldsymbol{U}^n_{i,j+1}) \end{aligned} \tag{9}$$

where $\lambda_A = (\Delta t S/V)\rho_A$. The solution to Eq. (9) is straightforward. On a scalar or vector computer Eq. (9) is solved using two sweeps. The first from the lower right to the upper left solving for δU^* by ignoring the negative flux Jacobians. Then δU is found by sweeping from upper right to lower left with δU^*, ignoring the positive flux Jacobians.

Solution of Eq. (9) on a massively parallel computer requires a modification to the LUSGS algorithm. Using sweeps on this type of computer would be inefficient because only a small fraction of the processors would be busy during the evaluation of the off-diagonal terms. The DP lower–upper relaxation (DPLUR) method[13] modifies the LUSGS algorithm in the following fashion. Instead of sweeps the equations are solved with a series of subiterations. These subiterations allow the computations to be performed in an almost perfect DP fashion. First, the right side of Eq. (8) is preconditioned to obtain $\delta U^{(0)}$:

$$\delta \boldsymbol{U}^{(0)}_{i,j} = (\boldsymbol{I} + \lambda^n_A \boldsymbol{I} + \lambda^n_B \boldsymbol{I})^{-1}_{i,j} \Delta \boldsymbol{U}_{i,j}$$

Then the $k_{\max}$ subiterations are made using the following:
For $k = 1, \quad k_{\max}$

$$\begin{aligned} &\delta \boldsymbol{U}^{(k)}_{i,j} = (\boldsymbol{I} + \lambda^n_A \boldsymbol{I} + \lambda^n_B \boldsymbol{I})^{-1}_{i,j} \{\Delta \boldsymbol{U}_{i,j} \\ &\quad + (\Delta t/V_{i,j})[\boldsymbol{A}^n_{+i-1/2,j} S_{i-1/2,j} \delta \boldsymbol{U}^{(k-1)}_{i-1,j} \\ &\quad - \boldsymbol{A}^n_{-i+1/2,j} S_{i+1/2,j} \delta \boldsymbol{U}^{(k-1)}_{i+1,j} + \boldsymbol{B}^n_{+i,j-1/2} S_{i,j-1/2} \delta \boldsymbol{U}^{(k-1)}_{i,j-1} \\ &\quad - \boldsymbol{B}^n_{-i,j+1/2} S_{i,j+1/2} \delta \boldsymbol{U}^{(k-1)}_{i,j+1}]\} \end{aligned} \tag{10}$$

then

$$\delta \boldsymbol{U}^{n+1}_{i,j} = \delta \boldsymbol{U}^{(k_{\max})}_{i,j}$$

With this approach, the required data for each subiteration have already been calculated in the previous subiteration. This allows for efficient operation as only nearest-neighbor communication is required. Additionally, the implicit flux vectors can be calculated and then communicated. This eliminates the need to communicate the entire flux Jacobian.

Results

The DP–TVD method has been implemented to solve several simple axisymmetric flows for which wind-tunnel data is available. Comparisons with this data will determine the validity of the method. Performance and accuracy issues are also addressed in a qualitative fashion with this preliminary data.

Flow Calculations

Supersonic flows over two simple geometries have been calculated with the TVD method. These geometries were also tested in wind tunnels in Refs. 16 and 17. The first of these geometries, model 7 in Ref. 16, is a cone–cylinder. The Mach number for this case was 2.96. The second geometry, model D in Ref. 17, is a simple body of revolution. The second case was run at a Mach number of 2.01. In an attempt to more closely approximate the wind-tunnel conditions, stings were approximated on the models. However, extensive information on the shape and size of the stings was not available, so that the representation is only an approximate one.

Pressure signatures from flows are compared with those measured in experiments. Figure 1 shows the nomenclature used in subsequent plots. Here, β is the Mach angle $\sqrt{M_\infty^2 - 1}$. The pressure is measured on a stationary boundary-layer bypass plate, while the model is moved in the flowfield. The angle of attack α is zero for all cases.

The first model tested was the cone cylinder. The reference length of the conical forebody was 2 in., while the cylindrical afterbody approximated the wind-tunnel sting. Figure 2 shows

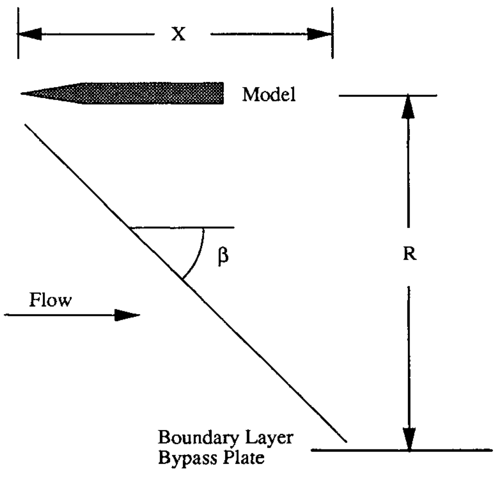

Fig. 1 Measurement nomenclature.

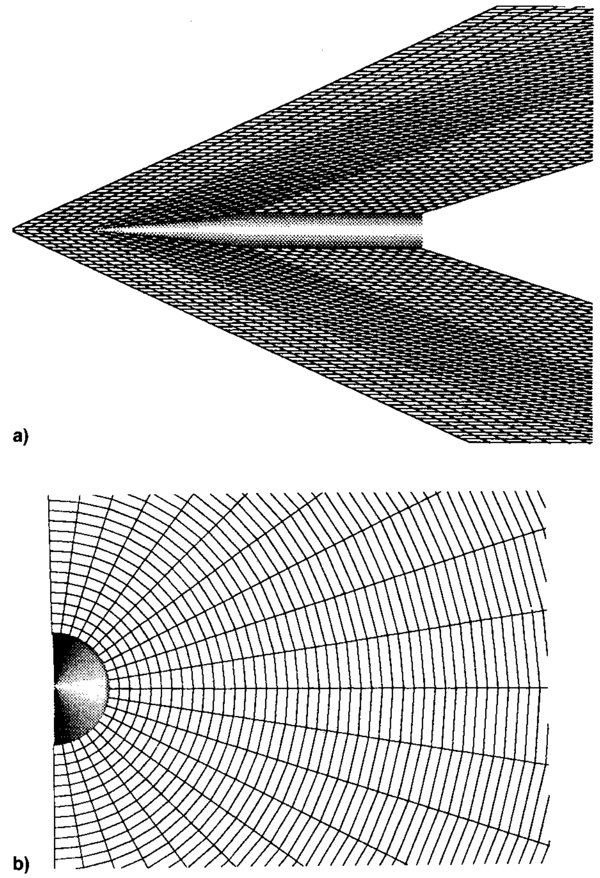

Fig. 2 Near-field a) symmetry and b) crossflow planes of the 32 × 256 × 12 computational mesh.

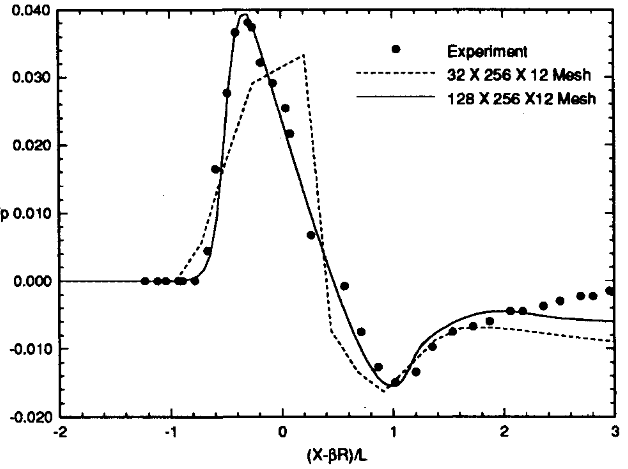

Fig. 3 Experimental and computed pressure signatures for the cone-cylinder body at five cone lengths normal to the body.

two views of the computational mesh about the model in the near field. For clarity only the near field of the mesh is shown. Additionally, since the flowfields are axisymmetric, calculations were only performed on the mesh in the top half of the figure.

Pressures were calculated at five body lengths normal to the body and compared with those found experimentally. The results are shown in Fig. 3. Two different computational grids were used in the calculations. It is evident from the figure that the 128 (axial) × 256 (radial) × 12 (azimuthal) grid provides suitable accuracy, while the 32 × 256 × 12 grid is not fine enough to resolve the flow accurately. Deviation from the experimentally measured pressures downstream is most likely due to the inaccuracies involved in modeling the shape and size of the wind-tunnel sting.

Near- and midfield pressure contours for the flow are plotted in Fig. 4. The bow shock and expansion region can be seen to be sharply resolved by the TVD method. In the midfield the shock waves can be seen to be propagting at the Mach angle.

The DP–TVD method was also used to calculate the flow about a parabolic body of revolution, model D in Ref. 16.

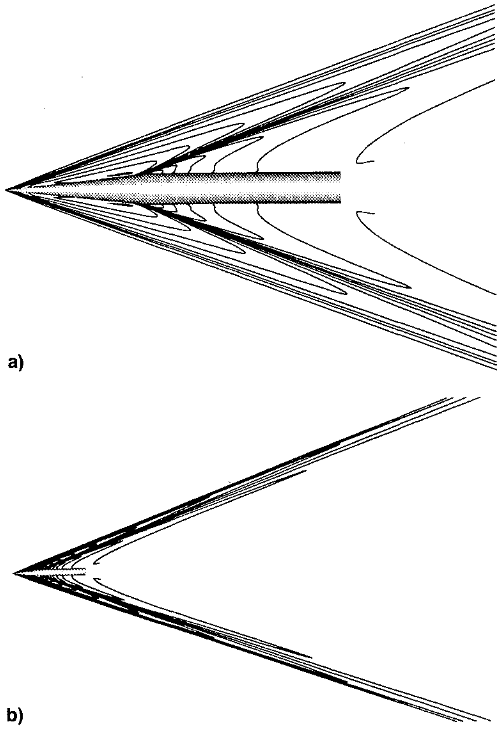

Fig. 4 a) Near-field and b) midfield pressure contours for the cone-cylinder at M = 2.96; 128 × 256 × 12 mesh.

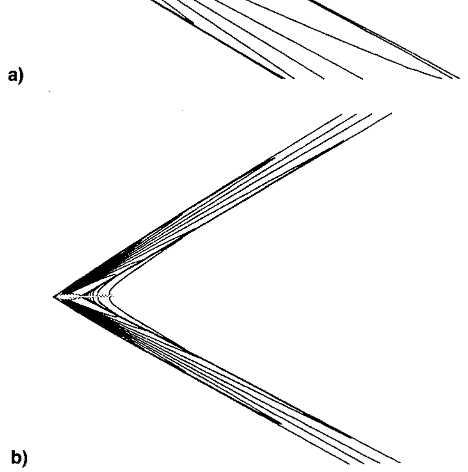

Fig. 5 a) Near-field and b) midfield pressure contours for the parabolic projectile at M = 2.01; 128 × 256 × 12 mesh.

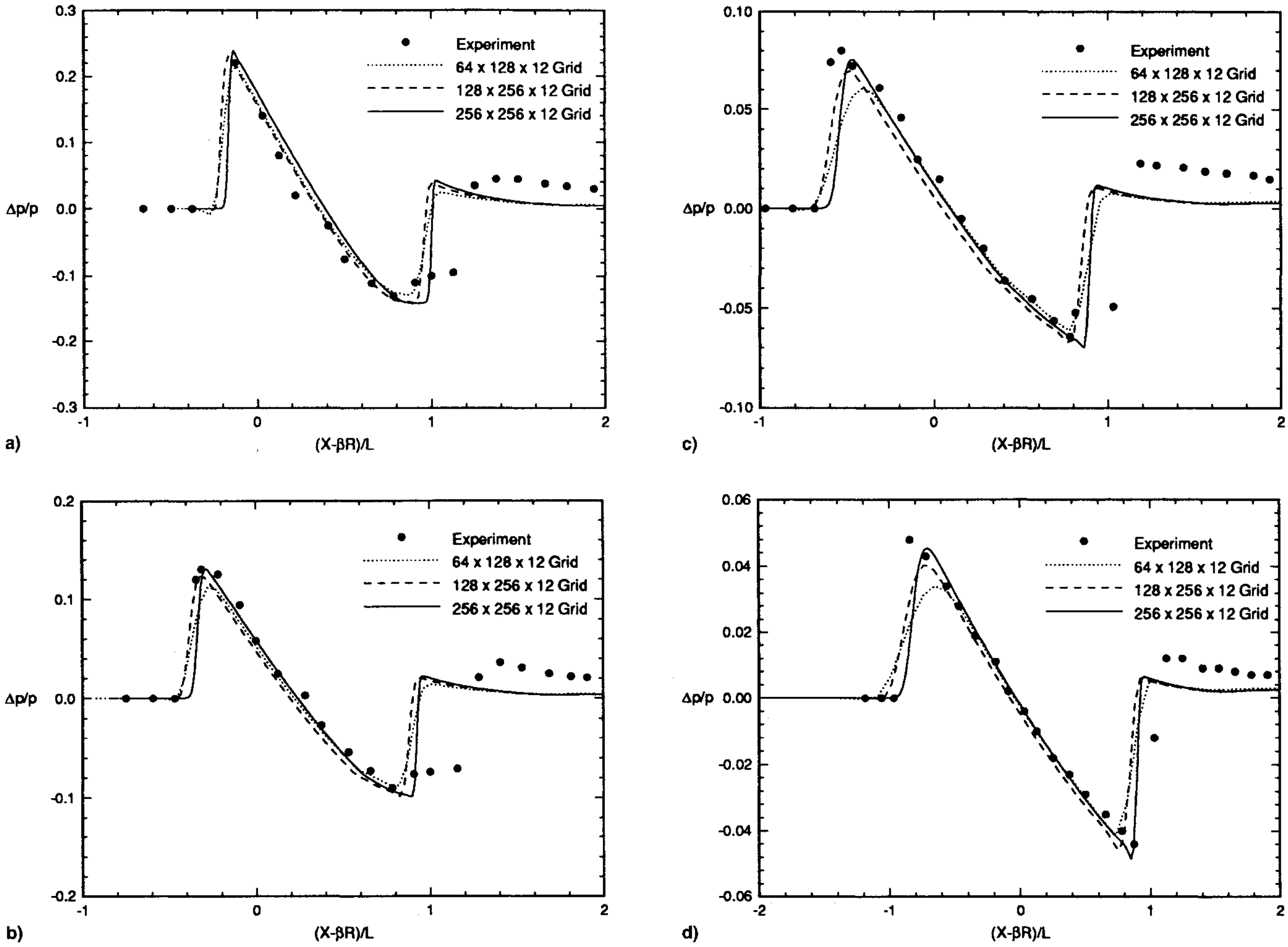

Fig. 6 Experimental and computed pressure signatures at H/L = a) 1.0, b) 2.0, c) 4.0, and d) 8.0 for the parabolic projectile.

Computational grids of varying sizes were used to determine the near- and midfield pressure signatures. Pressure contours in the symmetry planes are shown for Mach 2.01 flow in Fig. 5. A 128 × 256 × 12 mesh was used in these computations. The approximation used for the wind-tunnel sting has caused some discrepancies, as seen in the figure. The sting was approximated as one body length (2 in.) behind the body and 0.25 in. in diameter. This is most likely not an adequate approximation. Placing more cells in the region of the sting would help in resolving the flow more effectively.

Pressure signatures were extracted from the flowfield at four locations normal to the body axis. These signatures were then compared to the experimental data of Ref. 16. This data is presented in Fig. 6. It is evident that the DP–TVD scheme adequately resolves the shock structure in the near field for a relatively small amount of cells in the flow direction. Sixty-four cells seem to adequately resolve the bow shock and expansion region. However, in the midfield, a finer mesh is required. The best signature resolution came with the use of a 256 × 256 × 12 mesh. Even with this fine mesh there is still difficulty resolving the flow in the region of the wind-tunnel sting, but this appears to be due to the approximate nature of the sting definition in the calculations.

These two sets of calculations have shown the DP–TVD scheme to adequately predict the pressure signatures of supersonically moving objects in the near and midfields. Judicious mesh sizing and generation are essential for accurate results.

Performance Issues

The DP–TVD method has been shown to adequately predict the pressure signatures of supersonic flows, but the method must have excellent performance qualities for it to replace existing prediction methods. Some of the performance issues involved are discussed here.

Foremost among the performance issues is code size. The size of the code will determine the amount of cells that can be used in the computational domain. Since more cells lead to increased accuracy it is desirable to use as many as allowed by the available memory. Currently, a 256 × 256 × 12 mesh uses approximately 750 Mbyte, and fits easily on the CM-5's 64 processor partition. Thus, a mesh with over 6 million cells, such as a 512 × 512 × 24 mesh, could be employed on the 512 processor partition. This is a conservative estimate. With a dedicated partition even larger codes can be run. The available memory allows considerably higher number of cells than can currently be used with traditional prediction methods.

High floating point performance is also required of the new prediction method. Computational simulations are considerably more cost effective that wind-tunnel measurements, but an effort must be made to reduce the amount of time required for an adequate prediction. Thus, computational efficiency is essential. The program computed on a 128 × 256 × 12 mesh at approximately 1.7 Gflop on the CM-5's 64 processor partition. Similar values were obtained with similar meshes on 32 processors. If the problem size is increased accordingly, the floating point performance scales with an increase in the number of processors. Thus, a value of 14.0 Gflop would be expected on the 512 processor partition.

Codes of the size presented here usually converge to machine zero in several thousand iterations. This usually takes 30–45 min on a 32 or 64 processor partition, depending on the load level. Total CPU time is usually much less than this, on the order of 10–20 min. This is faster than most codes used in current prediction methods. Additional optimization efforts will increase performance further.

Conclusions and Future Directions

A new DP–TVD method for sonic boom calculations was introduced. The method is inherently parallel and can be utilized in massively parallel and distributed environments. The available memory of the parallel machines allows a considerably higher number of cells that can currently be used with traditional prediction methods. Since sonic boom calculations require a solution at comparatively high distances from the body, the number of cells is important, especially if detailed three-dimensional results are needed. This new method was shown to produce flow predictions efficiently for several sonic boom test cases. The method has the potential to be used as a primary tool in the evaluation of new HSCT configurations.

The DP–TVD method also has the ability to be utilized for the calculation of other types of supersonic and transonic flows. Future work in this research will focus on the prediction of the three-dimensional flowfields generated by supersonic aircraft. Efforts will be centered on increasing the accuracy of the method, while increasing performance when possible. Exploratory three-dimensional calculations are given in Ref. 11.

One method of increasing accuracy in the calculations is to use a nonuniform distribution of grid cells. This distribution will place more cells in areas of the flowfield, such as shock waves. The cells can be distributed nonuniformly if a priori knowledge of the flow solution exists. However, this knowledge is usually not available. A more applicable method is the solution-adaptive mesh generation algorithm of Benson and McRae.[18] In this method new locations are determined for the cell vertices based on weighting functions. The weighting functions are determined from the strength of flow gradients. The flow gradients come from an intermediate flow solution.

This mesh adaptation algorithm is preferable over others because it is simple to employ, is easily adaptable to parallel computations, and preserves the structured mesh required with the implicit algorithm. Current efforts involve reduction of the interprocessor communication involved in the algorithm. Some communication is necessary, but it should be minimized to reduce the effects on the overall performance.

Other additions may also be made to the current method so that it will be able to calculate viscous and possible chemical reaction effects. These effects dominate hypersonic re-entry flows and will be needed in flow predictions of a future aerospace plane of hypersonic transport. This method should work well for the calculation of other supersonic and transonic flows as well.

Acknowledgments

Support for this research was provided by the Army Research Office Contract DAALO3-0038 with the University of Minnesota Army High Performance Computing Research Center (AHPCRC) and the DoD Shared Resource Center at the AHPCRC. This work was also supported by allocation grants from the Minnesota Supercomputer Institute. The authors wish to thank Graham Candler and Michael Wright from the University of Minnesota for their assistance in developing the implicit portion of the DP–TVD code.

References

[1] Darden, C. M., and Shepherd, K. P., "Assessment and Design of Low Boom Configurations for Supersonic Transport Aircraft," DGLR/AIAA Paper 92-02-063, May 1992.

[2] Plotkin, K. J., "Review of Sonic Boom Theory," AIAA Paper 89-1105, April 1989.

[3] Whitham, G. B., "The Flow Pattern of a Supersonic Projectile," *Communications of Pure and Applied Mathematics*, Vol. V, No. 3, 1952, pp. 301–348.

[4] Walkden, F., "The Shock Pattern of a Wing Body Combination, Far from the Flight Path," *Aeronautical Quarterly*, Vol. IX, Pt. 2, May 1958, pp. 164–194.

[5] Darden, C. M., "Limitations of Linear Theory for Sonic Boom Calculations," *Journal of Aircraft*, Vol. 30, No. 3, 1993, pp. 309–314.

[6] Ferri, A., Ting, L., and Lo, R. W., "Nonlinear Sonic Boom Propagation Including the Asymmetric Effects," *AIAA Journal*, Vol. 15, No. 5, 1977, pp. 653–658.

[7] Siclari, M., and Darden, C., "A Euler Code Prediction of Near to Mid-Field Sonic Boom Pressure Signatures," AIAA Paper 90-4000, Oct. 1990.

[8] Cheung, S. H., Edwards, T. A., and Lawrence, S. L., "Application of CFD to Sonic Boom Near and Mid Flow-Field Prediction," *Journal of Aircraft*, Vol. 29, No. 5, 1992, pp. 920–926.

[9] Harten, A., "On a Class of High Resolution Total–Variation–Stable Finite Difference Schemes," *SIAM Journal of Numerical Analysis*, Vol. 21, No. 1, 1984, pp. 1–23.

[10] Yee, H. C., "A Class of High Resolution Explicit and Implicit Shock-Capturing Methods," NASA TM-101088, Feb. 1989.

[11] Pilon, A. R., and Lyrintzis, A. S., "A Data Parallel TVD Method for Sonic Boom Calculations," AIAA Paper 95-0833, Jan. 1995.

[12] Roe, P. L., "Approximate Riemann Solvers, Parameter Solvers, and Differencing Schemes," *Journal of Computational Physics*, Vol. 43, No. 2, 1981, pp. 357–372.

[13] Candler, G. V., Wright, M. J., and McDonald, J. D., "A Data Parallel Lower-Upper Relaxation Method for Reacting Flows," *AIAA Journal*, Vol. 32, No. 12, 1994, pp. 2380–2386.

[14] Yoon, S., and Jameson, A., "An LU-SSOR Scheme for the Euler and Navier-Stokes Equations," AIAA Paper 87-0600, Jan. 1987.

[15] MacCormack, R. W., and Candler, G. V., "The Solution of the Navier-Stokes Equations Using Gauss-Seidel Line Relaxation," *Computers and Fluids*, Vol. 17, No. 1, 1989, pp. 135–150.

[16] Carlson, H. W., "An Investigation of Some Aspects of the Sonic Boom by Means of Wind Tunnel Measurements of Pressures About Several Bodies at a Mach Number of 2.01," NASA TND-161, Dec. 1959.

[17] Shrout, B. L., Mack, R. J., and Dollyhigh, S. M., "A Wind Tunnel Investigation of Sonic-Boom Pressure Distributions of Bodies of Revolution at Mach 2.96, 3.83, and 4.63," NASA TND-6195, April 1971.

[18] Benson, R. A., and McRae, D. S., "A Solution Adaptive Mesh Algorithm for Dynamic/Static Refinement of Two and Three Dimensional Grids," *Proceedings: Numerical Grid Generation in CFD and Related Fields*, North-Holland, New York, 1991, pp. 185–200.