INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS, VOL. 10, 855–873 (1990)

NUMERICAL SIMULATION OF THREE-DIMENSIONAL TRANSONIC FLOWS

JUBARAJ SAHU

U.S. Army Ballistic Research Laboratory, LABCOM, Aberdeen Proving Ground, MD 21005–5066, U.S.A.

AND

JOSEPH L. STEGER

NASA Ames Research Center, Moffett Field, CA 94035, U.S.A.

SUMMARY

The three-dimensional flow over a projectile has been computed using an implicit, approximately factored, partially flux-split algorithm. A simple composite grid scheme has been developed in which a single grid is partitioned into a series of smaller grids for applications which require an external large memory device such as the SSD of the CRAY X-MP/48 or multi-tasking. The accuracy and stability of the composite grid scheme have been tested by numerically simulating the flow over an ellipsoid at an angle of attack and comparing the solution with a single-grid solution. The flow field over a projectile at $M = 0{\cdot}96$, 1·1 and 4° angle of attack has been computed using a fine grid and compared with experiment.

KEY WORDS 3D transonic flow Flux splitting Numerical simulation Composite grid

1. INTRODUCTION

The transonic flow over bodies of revolution is of considerable interest in aerodynamics. The critical aerodynamic behaviour occurs in this speed regime and a rapid change in the aerodynamic coefficients is observed. The flow field is complex owing to the presence of shocks and can be characterized by strong viscous–inviscid, shock–boundary layer interactions. At low transonic speeds, weak shock waves begin to form. As the speed is increased to $M = 1$, the shocks become larger and stronger. Further complications often occur in practical flows of interest owing to the presence of separated flow regions; this is especially true for flow at moderate to high angles of attack. As a result, it is desirable to use the Navier–Stokes computational technique to compute such flows.

In recent years Navier–Stokes[1–3] computational methods have been used to compute flow over bodies of revolution at transonic speeds. In Reference 1 the three-dimensional flow over a boat-tailed afterbody was computed where the primary emphasis was on the boat-tail flow field. Calculations of three-dimensional flow over a projectile have been made in Reference 2; however, the wake or base region flow was not computed due to lack of capability to compute such flows. The wake region has been included in a three-dimensional flow field computation in Reference 3. These calculations were made with the thin layer form of the compressible Navier–Stokes equations which were solved using the implicit Beam and Warming central finite difference scheme.[4–6] Generally, these calculations did not have sufficient grid resolution due to lack of

0271–2091/90/080855–19$09.50 *Received 19 July 1988*

adequate computer resources. With the advent of computers such as the CRAY X-MP/48 and CRAY 2 and the availability of the large memory (the core memory of the CRAY 2 and the fast external memory device SSD of the CRAY X-MP/48), it is now possible to use the fine computational grids needed for three-dimensional numerical simulation of transonic flows. Reference 7 is an excellent example where the CRAY 2 computer was used to perform transonic flow computations over a hemispherical cylinder at high angles of attack using its large in-core memory. In the present work we make extensive use of the SSD device of the CRAY X-MP/48 to provide sufficient grid resolution. This necessitates splitting the database of a large single grid into smaller pieces that fit within the in-core memory of the X-MP/48. While this break-up can be achieved in various ways, the simplest approach is to break the single grid into a number of smaller grids.

The numerical scheme used is an implicit scheme based on flux splitting[8] and upwind spatial differencing in the streamwise direction. Upwind schemes can have several adavantages over central difference schemes, including natural numerical dissipation and better stability properties.

The accuracy of the composite grid scheme has been tested by numerically simulating the transonic flow over an ellipsoid at 10° angle of attack and comparing this result with the result obtained with a single grid. In addition, the flow over a secant–ogive–cylinder–boat–tail projectile has been computed using a fine grid at $M = 0{\cdot}96$, $1{\cdot}1$ and 4° angle of attack. These computed results have been compared with experimental data to determine the accuracy of the numerical predictions. In Section 2 the governing equations, numerical procedure and composite grid scheme are described. Results are presented in Section 3.

2. NUMERICAL METHOD

Governing equations

The three-dimensional hyperbolic system of conservation equations of mass, momentum and energy can be represented in flux vector form as

$$\partial_\tau \hat{\mathbf{Q}} + \partial_\xi(\hat{\mathbf{F}} + \hat{\mathbf{F}}_v) + \partial_\eta(\hat{\mathbf{G}} + \hat{\mathbf{G}}_v) + \partial_\zeta(\hat{\mathbf{H}} + \hat{\mathbf{H}}_v) = \mathbf{0}, \tag{1}$$

where the independent variable τ is the time and the spatial independent variables ξ, η and ζ are chosen to map a curvilinear body-conforming discretization into a uniform computational space.

The general co-ordinate transformations

$$\begin{aligned} \xi &= \xi(x, y, z, t), \\ \eta &= \eta(x, y, z, t), \\ \zeta &= \zeta(x, y, z, t), \\ \tau &= t \end{aligned}$$

are used.

Here $\hat{\mathbf{Q}}$ contains all the dependent variables and $\hat{\mathbf{F}}$, $\hat{\mathbf{G}}$ and $\hat{\mathbf{H}}$ are the inviscid fluxes:

$$\hat{\mathbf{Q}} = J^{-1}\begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ e \end{bmatrix}, \qquad \hat{\mathbf{F}} = J^{-1}\begin{bmatrix} \rho U \\ \rho u U + \xi_x p \\ \rho v U + \xi_y p \\ \rho w U + \xi_z p \\ (e+p)U - \xi_t p \end{bmatrix},$$

$$\hat{\mathbf{G}} = J^{-1}\begin{bmatrix} \rho V \\ \rho u V + \eta_x p \\ \rho v V + \eta_y p \\ \rho w V + \eta_z p \\ (e+p)V - \eta_t p \end{bmatrix}, \qquad \hat{\mathbf{H}} = J^{-1}\begin{bmatrix} \rho W \\ \rho u W + \zeta_x p \\ \rho v W + \zeta_y p \\ \rho w W + \zeta_z p \\ (e+p)W - \zeta_t p \end{bmatrix}, \tag{2}$$

where J is the Jacobian of the co-ordinate transformation.

The flux terms $\hat{\mathbf{F}}_v$, $\hat{\mathbf{G}}_v$ and $\hat{\mathbf{H}}_v$ contain viscous derivatives, and throughout a non-dimensional form of the equations is used. The conservative form of the equations is maintained chiefly to capture the Rankine–Hugoniot shock jump relations as accurately as possible.

The Cartesian velocity components u, v and w are non-dimensionalized with respect to a_∞ (the free stream speed of sound), the density ρ is referenced to ρ_∞ and the total energy e to $\rho_\infty a_\infty^2$. Pressure is defined as

$$p = (\gamma - 1)[e - 0{\cdot}5\rho(u^2 + v^2 + w^2)], \tag{3}$$

where γ is the ratio of specific heats.

The contravariant velocity components are denoted by U, V and W and are related to the Cartesian components u, v and w by the relations

$$\begin{pmatrix} U - \xi_t \\ V - \eta_t \\ W - \zeta_t \end{pmatrix} = \begin{pmatrix} \xi_x & \xi_y & \xi_z \\ \eta_x & \eta_y & \eta_z \\ \zeta_x & \zeta_y & \zeta_z \end{pmatrix}\begin{pmatrix} u \\ v \\ w \end{pmatrix}. \tag{4}$$

Finally the metric terms are obtained from chain rule expansion of x_ξ, y_ξ, etc. and solved for ξ_x, ξ_y, etc. to give

$$\left.\begin{aligned} \xi_y &= J(y_\eta z_\zeta - x_\eta z_\zeta), & \eta_x &= J(z_\xi y_\zeta - y_\xi z_\zeta), \\ \xi_y &= J(z_\eta x_\zeta - x_\eta z_\zeta), & n_y &= J(x_\xi z_\zeta - x_\zeta z_\xi), \\ \xi_z &= J(x_\eta u_\zeta - y_\eta x_\zeta), & \eta_z &= J(y_\xi x_\zeta - x_\xi y_\zeta), \\ \zeta_x &= J(y_\zeta z_\eta - z_\xi y_\eta), & \xi_t &= -x_\tau \xi_x - y_\tau \xi_y - z_\tau \xi_z, \\ \zeta_u &= J(x_\eta z_\xi - x_\xi z_\eta), & \eta_t &= -x_\tau \eta_x - y_\tau \eta_y - z_\tau \eta_z, \\ \zeta_z &= J(x_\xi y_\eta - y_\xi x_\eta), & \zeta_t &= -x_\tau \zeta_x - y_\tau \zeta_y - z_\tau \zeta_z, \end{aligned}\right\} \tag{5}$$

with

$$J^{-1} = x_\xi y_\eta z_\zeta + x_\zeta y_\xi z_\eta + x_\eta y_\zeta z_\xi - x_\xi y_\zeta z_\eta - x_\eta y_\xi z_\zeta - x_\zeta y_\eta z_\xi$$

For body-conforming co-ordinates and high-Reynolds-number flow where ζ is the co-ordinate away from the surface, the thin layer approximation can be made in the ζ-direction. Such approximation is valid for attached and mildly separated flows as examined by Pulliam and Steger.[6] The governing equation (1) can then be written as

$$\partial_\tau \hat{\mathbf{Q}} + \partial_\xi \hat{\mathbf{F}} + \partial_\eta \hat{\mathbf{G}} + \partial_\zeta \hat{\mathbf{H}} = Re^{-1}\partial_\zeta \hat{\mathbf{S}}, \tag{6}$$

where

$$\hat{\mathbf{S}} = J^{-1}\begin{bmatrix} 0 \\ \mu(\zeta_x^2+\zeta_y^2+\zeta_z^2)u_\zeta + (\mu/3)(\zeta_x u_\zeta + \zeta_y v_\zeta + \zeta_z w_\zeta)\zeta_x \\ \mu(\zeta_x^2+\zeta_y^2+\zeta_z^2)v_\zeta + (\mu/3)(\zeta_x u_\zeta + \zeta_y v_\zeta + \zeta_z w_\zeta)\zeta_y \\ \mu(\zeta_x^2+\zeta_y^2+\zeta_z^2)w_\zeta + (\mu/3)(\zeta_x u_\zeta + \zeta_y v_\zeta + \zeta_z w_\zeta)\zeta_z \\ \{(\zeta_x^2+\zeta_y^2+\zeta_z^2)[0{\cdot}5\mu(u^2+v^2+w^2)_\zeta + \kappa Pr^{-1}(\gamma-1)^{-1}(a^2)_\zeta] \\ +(\mu/3)(\zeta_x u+\zeta_y v+\zeta_z w)(\zeta_x u_\zeta+\zeta_y v_\zeta+\zeta_z w_\zeta)\} \end{bmatrix} \tag{7}$$

Here κ is the coefficient of thermal conductivity, μ is the dynamic viscosity and Pr is the Prandtl number. The viscous terms in ζ have been collected into the vector $\hat{\mathbf{S}}$ and the non-dimensional reciprocal Reynolds number is extracted to indicate a viscous flux term.

In differencing these equations it is often advantageous to difference about a known base solution, denoted by subscript 0, as

$$\begin{aligned}\delta_\tau(\hat{\mathbf{Q}}-\hat{\mathbf{Q}}_0)+\delta_\xi(\hat{\mathbf{F}}-\hat{\mathbf{F}}_0)+\delta_\eta(\hat{\mathbf{G}}-\hat{\mathbf{G}}_0)+\delta_\zeta(\hat{\mathbf{H}}-\hat{\mathbf{H}}_0)-Re^{-1}\delta_\zeta(\hat{\mathbf{S}}-\hat{\mathbf{S}}_0)\\ =-\partial_\tau\hat{\mathbf{Q}}_0-\partial_\xi\hat{\mathbf{F}}_0-\partial_\eta\hat{\mathbf{G}}_0-\partial_\zeta\hat{\mathbf{H}}_0+Re^{-1}\partial_\zeta\hat{\mathbf{S}}_0,\end{aligned} \tag{8}$$

where δ indicates a general difference operator and ∂ is the differential operator. If the base state can be properly chosen, the differenced quantities can have smaller and smoother variation and therefore less differencing error. Although various choices are available for the base state, the free stream is used as a base solution in the present formulation and does satisfy the equation.

Implicit finite difference algorithm

For simplicity we begin with equation (6) which can be rewritten as

$$\partial_\tau\hat{\mathbf{Q}} = -(\partial_\xi\hat{\mathbf{F}}+\partial_\eta\hat{\mathbf{G}}+\partial_\zeta\hat{\mathbf{H}}-Re^{-1}\partial_\zeta\hat{\mathbf{S}}). \tag{9}$$

The time differencing is

$$\hat{\mathbf{Q}}^{n+1} = \hat{\mathbf{Q}}^n + \Delta t\left[\theta\left(\frac{\partial\hat{\mathbf{Q}}}{\partial t}\right)^{n+1} + (1-\theta)\left(\frac{\partial\hat{\mathbf{Q}}}{\partial t}\right)^n\right] + O(\Delta t^2), \tag{10}$$

where θ is a parameter which determines the type of scheme used. For example, $\theta = 1$ corresponds to the Euler implicit method whereas $\theta = \frac{1}{2}$ leads to the second-order time accurate Crank–Nicolson scheme. Substituting equation (9) into equation (10), one has

$$\begin{aligned}\hat{\mathbf{Q}}^{n+1} = \hat{\mathbf{Q}}^n - \Delta t[\theta(\partial_\xi\hat{\mathbf{F}}+\partial_\eta\hat{\mathbf{G}}+\partial_\zeta\hat{\mathbf{H}}-Re^{-1}\partial_\zeta\hat{\mathbf{S}})^{n+1}\\ +(1-\theta)(\partial_\xi\hat{\mathbf{F}}+\partial_\eta\hat{\mathbf{G}}+\partial_\zeta\hat{\mathbf{H}}-Re^{-1}\partial_\zeta\hat{\mathbf{S}})^n] + O(\Delta t^2).\end{aligned} \tag{11}$$

Equation (11) is non-linear since $\hat{\mathbf{F}}$, $\hat{\mathbf{G}}$, $\hat{\mathbf{H}}$ and $\hat{\mathbf{S}}$ are functions of the dependent variable $\hat{\mathbf{Q}}$. The non-linearity can be removed by a linearization procedure. As in the Beam–Warming algorithm,[4] the following time linearization is used:

$$\hat{\mathbf{F}}^{n+1} = \hat{\mathbf{F}}^n + \hat{\mathbf{A}}^n(\hat{\mathbf{Q}}^{n+1}-\hat{\mathbf{Q}}^n) + O(\Delta t^2), \tag{12}$$

where $\hat{\mathbf{A}}$ is a Jacobian matrix, $\hat{\mathbf{A}} = \partial\hat{\mathbf{F}}/\partial\hat{\mathbf{Q}}$. Similarly, the flux vectors $\hat{\mathbf{G}}$, $\hat{\mathbf{H}}$ and $\hat{\mathbf{S}}$ at the $(n+1)$th time level are linearized and result in Jacobian matrices $\hat{\mathbf{B}}$, $\hat{\mathbf{C}}$ and $\hat{\mathbf{M}}$ respectively.[6] Using the

above linearization, equation (11) can now be written in delta form ($\Delta\mathbf{Q}^n = \hat{\mathbf{Q}}^{n+1} - \hat{\mathbf{Q}}^n$):

$$[\mathbf{I} + \theta\Delta t(\delta_\xi \hat{\mathbf{A}}^n + \delta_\eta \hat{\mathbf{B}}^n + \delta_\zeta \hat{\mathbf{C}}^n - Re^{-1}\delta_\zeta J^{-1}\hat{\mathbf{M}}^n J)]\Delta\hat{\mathbf{Q}}^n$$
$$= -\Delta t[\delta_\xi \hat{\mathbf{F}}^n + \delta_\eta \hat{\mathbf{G}}^n + \delta_\zeta \hat{\mathbf{H}}^n - Re^{-1}\bar{\delta}_\zeta \hat{\mathbf{S}}^n], \tag{13}$$

where δ is a second-order accurate spatial difference operator and $\bar{\delta}$ is a midpoint operator. Using approximate factorization, the left-hand-side operator of equation (13) can be factored into three factors, one in each direction. In addition, if δ is a central finite difference operator, then we have the standard Beam and Warming scheme. Alternately, one can construct the two-factored algorithm proposed by Steger and Buning[9] which consists of upwind differencing in the streamwise direction and central differencing in the other two directions. Such a scheme obtained from equation (13) is as follows:

$$[\mathbf{I} + \theta\Delta t(\delta_\xi^b \hat{\mathbf{A}}^n + \delta_\eta^c \hat{\mathbf{B}}^n)][\mathbf{I} + \theta\Delta t(\delta_\zeta^c \hat{\mathbf{C}}^n - Re^{-1}\bar{\delta}_\zeta J^{-1}\hat{\mathbf{M}}^n J)]\Delta\hat{\mathbf{Q}}^n = \text{RHS}, \tag{14}$$

where RHS is the right-hand side of equation (13):

$$\text{RHS} = -\Delta t\,\delta_\xi \hat{\mathbf{F}}^n + \delta_\eta \hat{\mathbf{G}}^n + \delta_\zeta \hat{\mathbf{H}}^n - Re^{-1}\delta_\zeta \hat{\mathbf{S}}^n.$$

A variant of this scheme is one where flux vector splitting is used in the streamwise ξ-direction. If $\hat{\mathbf{F}}$ satisfies the homogeneous property $\hat{\mathbf{F}} = \hat{\mathbf{A}}\hat{\mathbf{Q}}$ and $\hat{\mathbf{A}}$ has a complete set of linearly independent eigenvectors, then the flux vector $\hat{\mathbf{F}}$ can be split into two subvectors. The eigenvalues associated with $\hat{\mathbf{F}}^+$ are all positive and those associated with $\hat{\mathbf{F}}^-$ all negative, i.e. approximate backward and forward upwind differencing can then be used with these subvectors $\hat{\mathbf{F}}^+$ and $\hat{\mathbf{F}}^-$ respectively. The flux vector $\hat{\mathbf{F}}$ is thus split as[8]

$$\hat{\mathbf{F}} = \hat{\mathbf{F}}^+ + \hat{\mathbf{F}}^-. \tag{15}$$

$\hat{\mathbf{F}} = \hat{\mathbf{A}}\hat{\mathbf{Q}} = \mathbf{x}\Lambda\mathbf{x}^{-1}\hat{\mathbf{Q}}$, where Λ is a diagonal matrix which can be split into Λ^+ and Λ^- based on positive and negative eigenvalues respectively.

$$\hat{\mathbf{F}} = \mathbf{x}(\Lambda^+ + \Lambda^-)\mathbf{x}^{-1}\hat{\mathbf{Q}} = (\hat{\mathbf{A}}^+ + \hat{\mathbf{A}}^-)\hat{\mathbf{Q}}. \tag{16}$$

The Jacobian matrix $\hat{\mathbf{A}}$ is thus split as

$$\hat{\mathbf{A}} = \hat{\mathbf{A}}^+ + \hat{\mathbf{A}}^-.$$

In general co-ordinates the five eigenvalues of $\hat{\mathbf{A}}$ are given by

$$\lambda(\hat{\mathbf{A}}) = (\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5)^T, \tag{17}$$

where

$$\lambda_1 = \lambda_2 = \lambda_3 = U,$$
$$\lambda_4 = U + c\sqrt{(\xi_x^2 + \xi_y^2 + \xi_z^2)},$$
$$\lambda_5 = U - c\sqrt{(\xi_x^2 + \xi_y^2 + \xi_z^2)}$$

and $c = \sqrt{(\gamma p/\rho)}$.

Using the above flux vector splitting, a two-factored algorithm similar to equation (14) can be constructed:

$$(\mathbf{I} + \theta\Delta t\mathscr{L}^+)(\mathbf{I} + \theta\Delta t\mathscr{L}^-)\Delta\hat{\mathbf{Q}}^n = -\Delta t\,\delta_\xi^b(\hat{\mathbf{F}}^+)^n + \delta_\xi^f(\hat{\mathbf{F}}^-)^n + \delta_\eta \hat{\mathbf{G}}^n + \delta_\zeta \hat{\mathbf{H}}^n - Re^{-1}\bar{\delta}_\zeta \hat{\mathbf{S}}^n, \tag{18}$$

where the operators $\mathscr{L}^+$ and $\mathscr{L}^-$ are given by

$$\mathscr{L}^+ = \delta_\xi^b(\hat{\mathbf{A}}^+)^n + \delta_\zeta \hat{\mathbf{C}}^n - Re^{-1}\bar{\delta}_\zeta J^{-1}\hat{\mathbf{M}}^n J,$$
$$\mathscr{L}^- = \delta_\xi^f(\hat{\mathbf{A}}^-)^n + \delta_\eta \hat{\mathbf{B}}^n. \tag{19}$$

The operator $(\mathbf{I} + \theta\Delta t\mathscr{L}^+)$ is a lower triangular matrix containing a second-order backward difference operator (δ_ξ^b) in ξ and a central difference operator (δ_ζ) in ζ. The other operator $(\mathbf{I} + \theta\Delta t\mathscr{L}^-)$ is an upper triangular matrix containing a second-order forward difference operator (d_ξ^f) in ξ and a central difference operator (δ_η) in η. Typically, δ_ξ^f and δ_ξ^b are three-point difference operators and $\bar{\delta}$ is the midpoint operator used with the viscous terms.

The final form of the implicit two-factored flux-split algorithm used in this study is given as

$$\begin{aligned}[\mathbf{I} + h\delta_\xi^b(\hat{\mathbf{A}}^+)^n + h\delta_\zeta\hat{\mathbf{C}}^n - hRe^{-1}\bar{\delta}_\zeta J^{-1}\hat{M}^n J - D_i|_\zeta][\mathbf{I} + h\delta_\xi^f(\hat{\mathbf{A}}^-)^n + h\delta_\eta\hat{B}^n - D_i|_\eta]\Delta\hat{\mathbf{Q}}^n \\ = -\Delta t\{\delta_\xi^b[(\hat{\mathbf{F}}^+)^n - \hat{\mathbf{F}}_\infty^+] + \delta_\xi^f[(\hat{\mathbf{F}}^-)^n - \hat{\mathbf{F}}_\infty^-] + \delta_\eta(\hat{\mathbf{G}}^n - \hat{\mathbf{G}}_\infty) \\ + \delta_\zeta(\hat{\mathbf{H}}^n - \hat{\mathbf{H}}_\infty) - Re^{-1}\bar{\delta}_\zeta(\hat{\mathbf{S}}^n - \hat{\mathbf{S}}_\infty)\} - D_e(\hat{\mathbf{Q}}^n - \hat{\mathbf{Q}}_\infty),\end{aligned} \tag{20}$$

where $h = \Delta t$ or $\Delta t/2$ and the free stream base solution is used. Artificial dissipation operators D_e and D_i are used in the central space differencing directions. The added artificial dissipation is a combination of second-order as well as fourth-order terms. To eliminate oscillations near shocks, it is desirable to use second-order numerical dissipation locally near shocks and fourth-order dissipation elsewhere. These dissipation terms are of the form

$$\begin{aligned}D_e\Big|_\eta &= (\Delta t)J^{-1}\varepsilon_2\bar{\delta}\rho(\hat{\mathbf{B}})\beta\bar{\delta} + \varepsilon_4\bar{\delta}\frac{\rho(\hat{\mathbf{B}})}{1+\beta}\bar{\delta}^3\Big|_\eta J, \\ D_i\Big|_\eta &= (\Delta t)J^{-1}\varepsilon_2\bar{\delta}\rho(\hat{\mathbf{B}})\beta\bar{\delta} + 2{\cdot}5\varepsilon_4\bar{\delta}\frac{\rho(\hat{\mathbf{B}})}{1+\beta}\bar{\delta}^3\Big|_\eta J,\end{aligned} \tag{21}$$

where $\rho(\hat{\mathbf{B}})$ is the spectral radius of $\hat{\mathbf{B}}$ and

$$\beta = \left|\frac{\bar{\delta}^2 P}{(1+\bar{\delta}^2)P}\right|.$$

Here P is the non-dimensional pressure, ε_2 is $O(1)$ and ε_4 is $O(0{\cdot}1)$. Near high-pressure-gradient regions such as shocks, β becomes large; thus second-order dissipation is used while the fourth-order dissipation is reduced. Similar artificial dissipation terms are also used in the ζ-direction. The implicit dissipation D_i works on $\Delta\hat{\mathbf{Q}}$ and thus accuracy is not impaired.

The factored left-hand-side operators of equation (20) can be readily inverted by sweeping and inversion of tridiagonal matrices with 5×5 blocks. This two-factor implicit scheme is readily vectorized or multi-tasked in planes of ξ = constant.

Composite grid scheme

In the present work a composite grid scheme has been developed in which a large single grid is split into a number of smaller grids, so that each of these grids can use the available core memory in turn while the rest are stored on the SSD device. The use of such a composite scheme, however, requires special care in storing and fetching the interface boundary data.

If a code is well written in a modular form for a single grid, it is a relatively simple task to reconfigure it either as a composite grid code or a blocked grid data structure code using an external large memory device such as the SSD of the CRAY X-MP/48. The numerical scheme is not a highly sensitive issue. The simplest case to begin with is the blocked grid data structure code in which a single large grid is partitioned into a series of smaller grid blocks as sketched in Figure 1. Such a partitioning (or domain decomposition) of a single original grid may come about because all of the single grid will not fit into a particular computer's high-speed memory or geometric constraints, or because a block iterative technique is in use. Partitioning can also be

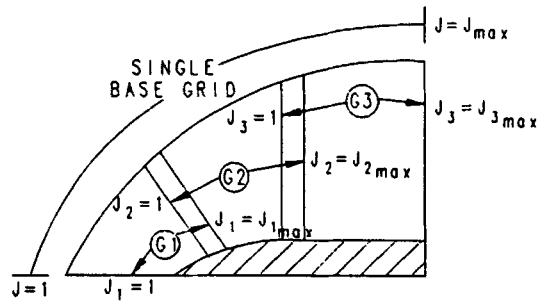

Figure 1. Single grid partitioned into smaller grids

influenced by different flow regimes, and in fact different numerical schemes can be used in different blocks. To make a code that works for a single grid work for a partitioned grid, it is necessary to build and verify an external data management scheme for a block update process (e.g. block iterative or block time-dependent process). In a block update scheme, if data from one subgrid are used to update the interface boundary of another subgrid, the sub-block solution will approach the solution of the single grid. (We say 'approach' because the differencing approximations near a boundary may drop in order of accuracy or use metrics evaluated in a different manner. Likewise, iterative convergence may be less tight in a given block.)

A code developed for a single grid can be made to work for a block grid structure by (1) mapping and storing the information for each grid onto a large memory and (2) supplying interface boundary arrays, pointers and updating procedures. Consider the situation in Figure 1 in which the single grid from $J = 1$ to $J_{max}$is partitioned into three grids. Although not shown in this figure, the base (or wake) region of a projectile is easily included by adding one or more zones in that region. We begin the procedure with grid G1. For each grid an estimated solution (or initial quess) is put into the large memory along with all pertinent information such as the grid, metrics, M_∞, Re, etc. These data are brought into the working memory and appropriate boundary update information is also called in and used to update any interface boundaries. At this point only the problem of dealing with a single-grid code must be dealt with. Solution variables are updated as before (more than one iterative update per subgrid is possible or interface boundaries can be iteratively corrected). The updated solution is then stored back into large (external) memory. Also stored into the appropriate locations is any information needed to update other interface boundaries. We then proceed to the next subgrid and so on.

The main problem of any blocked or composite grid scheme is the problem of storing and fetching the interface boundary data from its appropriate place, i.e. setting up the proper bookkeeping. Additional complications come about when we try to make blocked grids with interface boundaries give a solution that is identical to the solution obtained by a single large grid with no interface boundaries. This involves maintaining the same or equivalently good metrics, differencing accuracy, conservation statements, etc. at or near the interface boundaries that would exist on a continuous smooth grid. One choice which allowed simpler management of the interface boundary data while still maintaining second-order accuracy is described below.

Three-point backward and forward difference operators are used at the interior points. Near the interface, for example, at $J_2 = J_{2_{max}} - 1$, the three-point forward difference operator cannot be used with the one-grid-point overlap used and as shown in Figure 1.The differencing accuracy can be dropped from second-order to first-order; however, as will be shown in the results, this leads to inaccuracies in the flow field solution near the interfaces. To maintain second-order

accuracy near the interfaces, we difference, for example, $\partial \mathbf{F}/\partial \xi$ at J_2 and $J_{2_{\max}} - 1$ as

$$\frac{\partial \mathbf{F}}{\partial \xi} = \partial_\xi^{\mathrm{b}}(\mathbf{F}^+) + \partial_\xi^{\mathrm{f}}(\mathbf{F}^-),$$

where $\partial_\xi^{\mathrm{b}}$ is the usual three-point backward difference operator and $\partial_\xi^{\mathrm{f}}$ is now a central difference operator, i.e.

$$\frac{\partial \mathbf{F}}{\partial \xi} = \frac{3\mathbf{F}_J^+ - 4\mathbf{F}_{J-1}^+ + \mathbf{F}_{J-2}^+}{2\Delta\xi} + \frac{\mathbf{F}_{J+1}^- - \mathbf{F}_{J-1}^-}{2\Delta\xi}. \tag{22}$$

Near the other interface of grid G2($J_2 = 2$), the corresponding $\partial_\xi^{\mathrm{b}}$ operator is replaced by a central difference operator while $\partial_\xi^{\mathrm{f}}$ is the usual three-point forward difference operator. The planes $J_2 = 1$ and $J_2 = J_{2_{\max}}$ are, of course, boundaries for grid G2 and get their data from interior flow field solutions from neighbouring grids. Second-order accuracy at and near the interfaces is thus maintained. Partial use of central differencing near the interfaces has not adversely affected the stability of the scheme.

For the simple partitioning shown in Figure 1, all subgrid points are members of the original grid and no interpolation is required. This procedure thus has the advantage over patched or overset grid schemes which need interpolations. The partitioned grid has four interface boundaries: $J_1 = J_{1_{\max}}$, $J_2 = 1$, $J_2 = J_{2_{\max}}$ and $J_3 = 1$. Data for these planes are to be supplied from the other grids by injecting interior values of the other grids onto the interface boundaries. For example, the $J_1 = J_{1_{\max}}$ boundary of G1 may be taken as the $J_2 = 2$ or $J_2 = 3$ plane interior values of G2. More overlap would mean more implicit-like behaviour since more exchange of information would take place. Because only one grid and some boundary and pointer arrays can be in the working memory at one time, a low-storage scheme to shuffle data from one grid to another must be used. In working on grid G1, all of the usual data **Q**, x, y, z, . . . of grid G1 plus the interface boundary data stored in an array, say **QB**, from grids G2, G3, . . . will have to be in working memory. The array **QB** need only store **Q** on the interface boundary since other data such as metrics on the interface boundary can be computed from the usual data x, y, z, ... After updating grid G1, interface boundary data needed by the other grids from G1 should be loaded into the array **QB** while all data of G1 are still in memory. The data array **QB** should be shuffled back into a permanent large external memory, say **QBC**, in such a way that it can be readily brought back for grids G2, G3, etc. There are several ways to store **QB** in memory; the approach selected here is flexible and is used in the Chimera code.[10]

Other computational aspects

As pointed out earlier, $\theta = \frac{1}{2}$ or 1 in equation (18) or $h = \Delta t/2$ or Δt in equation (20) depends on the time accuracy of the scheme. $\theta = \frac{1}{2}$ is used for second-order accuracy in time (Crank–Nicolson method). For our present computatins the final steady state is the desired result. Thus a first-order time accurate Euler implicit scheme ($\theta = 1$) is used. To accelerate convergence one can also use time step sequences or spatially variable time steps. For a fixed Δt the Courant number is not uniform since the grid spacings usually vary from very fine to very coarse in the flow field domain of interest. The use of a space-varying Δt can thus be interpreted as an attempt to use a more uniform Courant number throughout the field.

For an aerodynamic simulation where the grid is highly stretched, we can use a purely geometric variation of Δt given by

$$\Delta t = \frac{\Delta t_{\mathrm{ref}}}{1 + \sqrt{J}}, \tag{23}$$

where Δt_{ref} is a reference-fixed Δt. Such scaling has been used by other researchers.[11,12] The time step h in equation (20) is then replaced by Δt given in equation (23).

The typical boundary conditions required for our problems consist of:

(a) the body surface at the $\zeta = 1$ plane
(b) free stream conditions at the $\zeta = \zeta_{max}$ plane
(c) axis conditions at the $\psi = 1$ plane
(d) extrapolation at the downstream boundary ($\xi = \xi_{max}$)
(e) periodic condition of the cross-flow plane (η) for the full 3D case or symmetry condition for the half 3D case.

The no-slip boundary condition for viscous flow is enforced by setting the contravariant velocities to zero, i.e.

$$U = V = W = 0. \tag{24}$$

The Cartesian components are obtained by the inverse of equation (4) or

$$\begin{pmatrix} u \\ v \\ w \end{pmatrix} = \begin{pmatrix} \xi_x & \xi_y & \xi_z \\ \eta_x & \eta_y & \eta_z \\ \zeta_x & \zeta_y & \zeta_z \end{pmatrix}^{-1} \begin{pmatrix} U - \xi_t \\ V - \eta_t \\ W - \zeta_t \end{pmatrix}. \tag{25}$$

The density at the body surface is usually obtained by a zeroth-order extrapolation from the interior of the flow field for adiabatic conditions. The pressure on the body surface is computed from the normal momentum equation[6]

$$(\nabla\xi \cdot \nabla\zeta)P_\xi + (\nabla\eta \cdot \nabla\zeta)P_\eta + (\nabla\zeta \cdot \nabla\zeta)P_\zeta = \rho(\partial_\tau \zeta_t + u(\nabla\zeta)_\tau) - \rho U(\nabla\zeta \cdot u_\xi) - \rho V(\nabla\zeta \cdot u_\eta) \tag{26}$$

which is solved numerically. The variables on the axis are obtained by averaging the extrapolated values from the interior points around the axis. Dirichlet boundary conditions are used at the free stream or far field boundary.

For the computation of turbulent flows, a turbulence model must be supplied. In the present calculations for transonic flow past a projectile, a two-layer algebraic eddy viscosity model due to Baldwin and Lomax[13] is used. In their two-layer model, the inner region follows the Prandtl–Van Driest formulation. The outer formulation can be used in attached and mildly separated boundary layers, and with minor modifications in wakes as well. In both the inner and outer formulations the distribution of vorticity is used to determine the length scales, thereby avoiding the necessity of finding the outer edge of the boundary layer (or wake). This simple model is not strictly valid for the complicated wake flows that exist behind the bluff base of a projectile. Higher-order turbulence models may lead to improved accuracy in this area.

3. RESULTS

The implicit time-marching procedure was used to obtain the desired steady state result starting from initial free stream conditions everywhere Boundary conditions were updated explicitly at each time step. The l_2 norm of the residual dropped at least three orders of magnitude before converged solutions were obtained. In addition, the surface pressure distribution was checked for time invariance. Solutions have been obtained for two cases: (i) flow over an ellipsoid at $M_\infty = 0{\cdot}90$ and $\alpha = 10°$, primarily to check the accuracy of the composite grid solution; and (ii) flow over a projectile at $M_\infty = 0{\cdot}96$, $1{\cdot}1$ and $\alpha = 4°$ which is calculated using a fine grid.

First, the flow over an ellipsoid at $M_\infty = 0{\cdot}90$ and 10° angle of attack was considered. Figure 2 shows the computational grid for the ellipsoid. Figure 2(a) shows the surface of the ellipsoid while Figure 2(b) is a cross-section of the full grid showing the longitudinal grid point distribution. This grid has 41 points in the streamwise direction, 18 points in the circumferential direction and 31

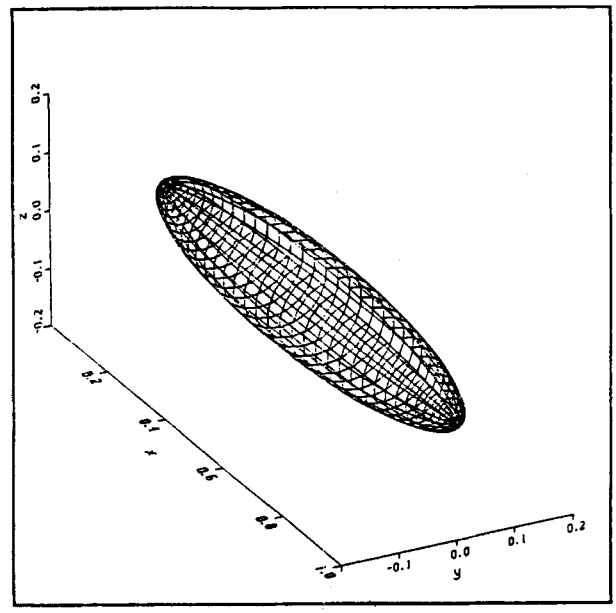

Figure 2(a). Surface grid for the ellipsoid

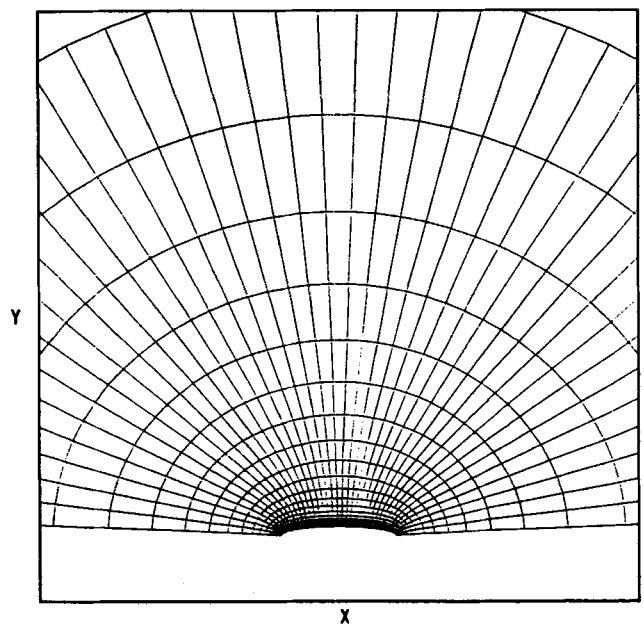

Figure 2(b). Longitudinal cross-section of the computational grid

points in the normal direction. Such a small grid was chosen so that the full grid solution could be obtained using a CRAY X-MP/48 with a two-million-word central core memory. The full grid was then partitioned into three smaller grids and the SSD device was used to obtain the solution for this composite grid scheme. Figure 3 shows the surface pressure distribution on the lee side. Here the full grid solution is shown as a solid line and the composite grid solution as a dashed line. The differences in these curves are apparent and occur near $X = 0{\cdot}3$ and $0{\cdot}8$ at the interfaces between the partitioned grids. The spatial accuracy at the interface was first-order in this case. Improving the spatial differencing accuracy at these boundaries to second-order resulted in the solution shown in Figure 4. As shown in this figure, the composite grid solution is virtually identical to the full grid solution.

Second, the flow over a projectile at a transonic speed $M_\infty = 0{\cdot}96$ and at a higher transonic speed $M_\infty = 1{\cdot}1$ for $\alpha = 4°$ was computed. The model used for the experiment and computational study presented here is an idealization of a realistic artillery projectile geometry. The experimental model shown in Figure 5 consists of a three-calibre (one calibre = maximum body diameter), sharp, secant–ogive nose, a two-calibre cylindrical mid-section and a one-calibre 7° conical afterbody or boat-tail. A similar model was used for the computational studies with the only difference being a 5% rounding of the nose tip. The nose tip rounding was done for computational efficiency and is considered to have little impact on the final integrated forces.

Experimental pressure data[14, 15] are available for this shape and were obtained in the NASA Langley eight-foot Pressure Tunnel using a sting-mounted model. The test conditions of 1 atm supply pressure and 320 K supply temperature resulted in a Reynolds number of $4{\cdot}5 \times 10^6$ based on model length.

The computational grid used is shown in Figure 6. Figure 6(a) shows the longitudinal cross-section of the 3D grid while Figure 6(b) shows the circumferential distribution of grid points. The grid consists of 162 points in the streamwise direction, 36 points in the circumferential direction and 50 points in the normal direction. This amounts to a little over twelve million words of storage for the code on the CRAY X-MP/48. Only up to 4 Mw of central core memory was easily

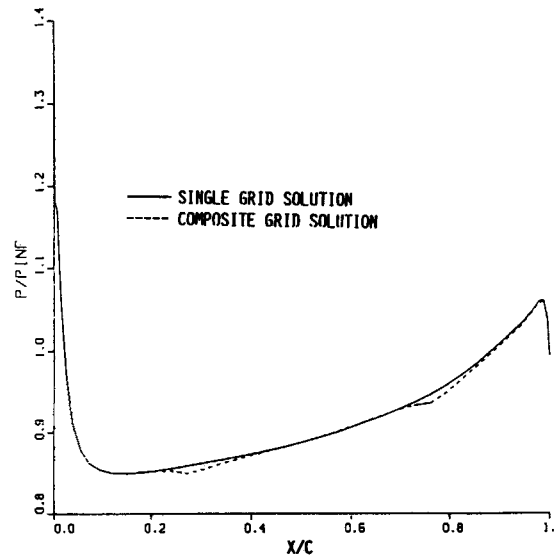

Figure 3. Surface pressure distribution, lee side; $M_\infty = 0{\cdot}90$, $\alpha = 10°$ (first-order accuracy at interface)

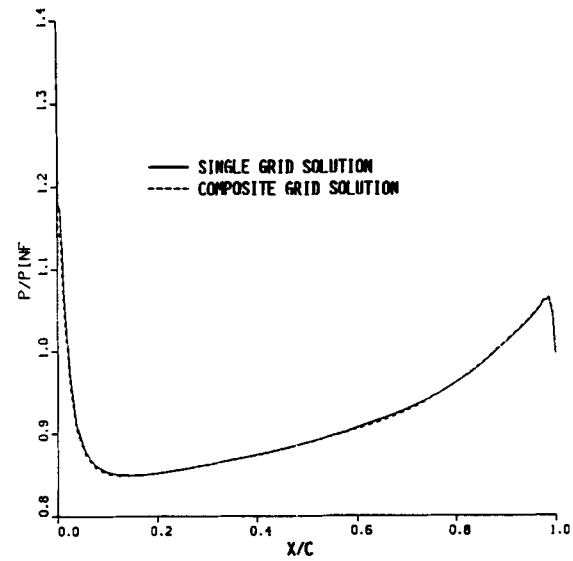

Figure 4. Surface pressure distribution, lee side; $M_\infty = 0{\cdot}90$, $\alpha = 10°$ (second-order accuracy at interface)

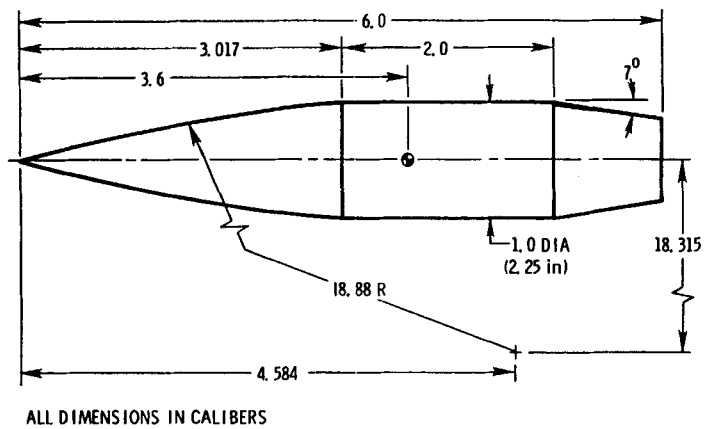

Figure 5. Model geometry of a projectile

accessible; therefore the full grid was partitioned into four smaller grids, each of which would use the core memory in turn while the rest was stored on the SSD device. This computation was performed on the CRAY X-MP/48 at the US Army Ballistic Research Laboratory. As shown in Figure 5, the clustering of grid points near the body surface is done to resolve the viscous boundary layer near the body surface. Grid clustering has also been used in the longitudinal direction near the boat-tail and the base corners where large gradients in the flow variables are expected. In addition, the grid is made to wrap around the base corner to facilitate the generation of the large single grid. What affect it has on the base region flow is currently under investigation and has not been studied here. The primary emphasis in this research was on the use of the composite grid technique itself.

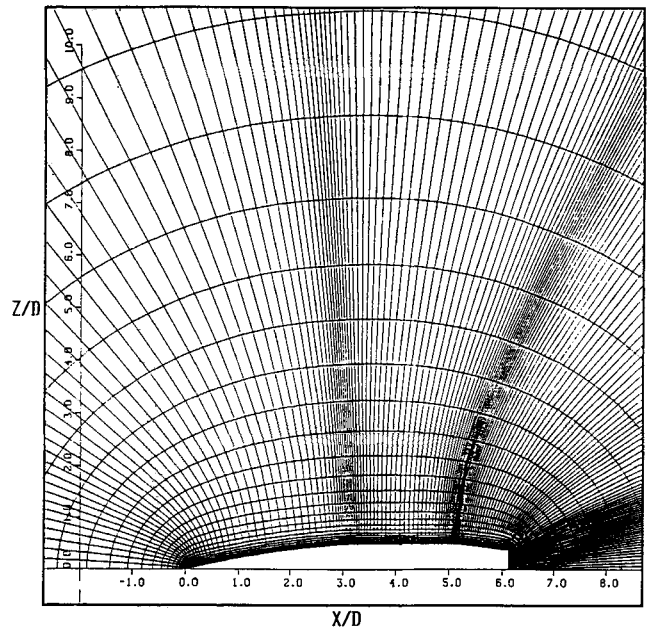

Figure 6(a). Computational grid (longitudinal cross-section)

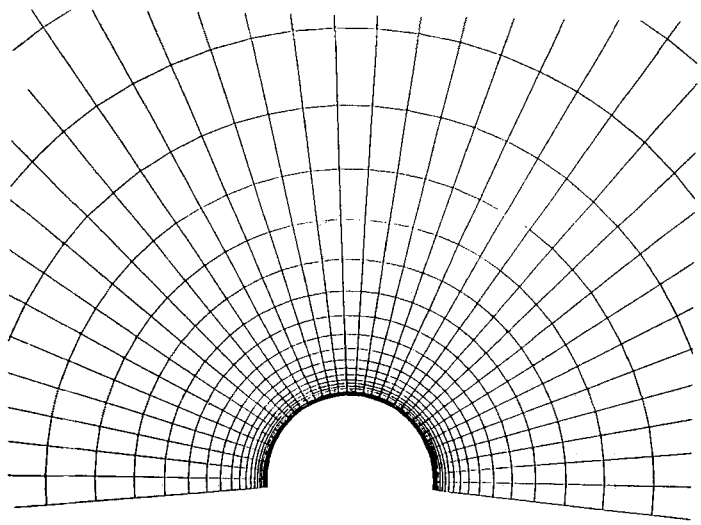

Figure 6(b). Computational grid (circumferential distribution)

Figures 7 and 8 show the pressure contours and Mach contours for the projectile in the windward and leeward planes. As seen in Figure 7, the pressure contours show the expansions at the ogive–cylinder, boat-tail and base corners. The expansions at the base corners are rather weak. This figure also indicates the presence of shock waves on the cylinder and on the boat-tail.

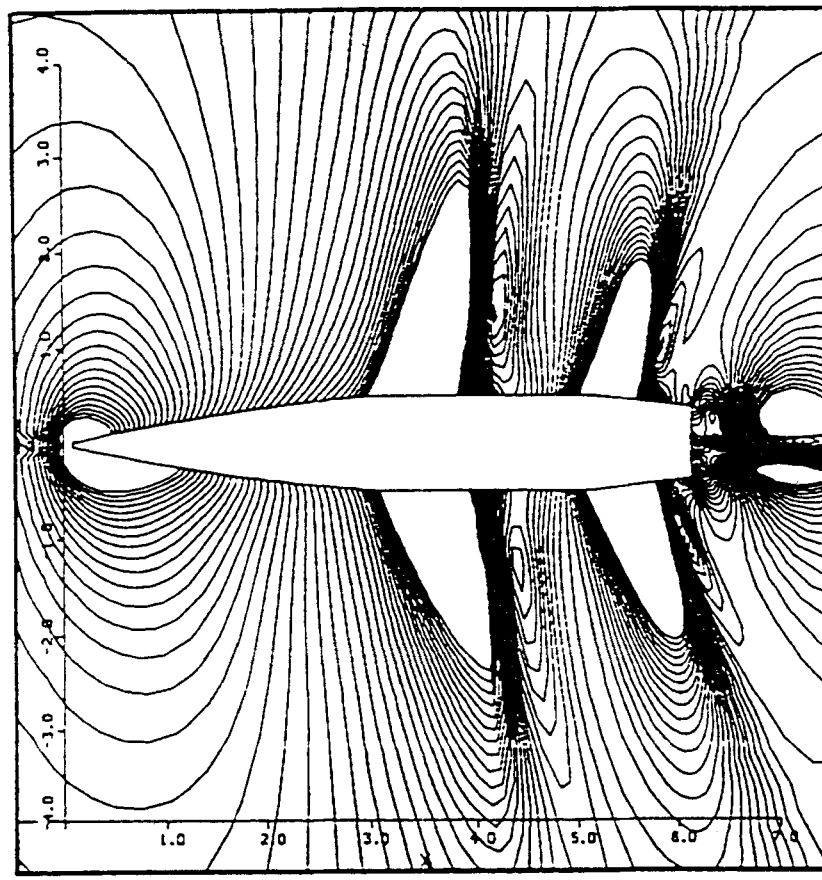

Figure 7. Pressure contours; $M_\infty = 0{\cdot}96$, $\alpha = 4°$

Figure 8 shows the Mach number contours. It clearly shows the expansions and shock structure that typically occur on the projectile at transonic speeds. Sharp shocks are clearly observed on the boat-tailed flow field which are also asymmetrically located (the one on the wind side being closer to the base than its counterpart on the lee side.) This figure also indicates the presence of strong free shear layers in the base region flow field and shows the expected asymmetric flow pattern in the wake. The asymmetric wake flow can be better seen in Figure 9 which shows the velocity field in the near wake for both the wind as well as the lee side. The recirculatory flow pattern in the base region is evident. Three separated flow bubbles can be clearly seen in this figure. The experimental model was sting-mounted in the base region and current work is in progress to model the sting before accuracy of the base region flow prediction can be determined. In the present computation the sting has not been modelled.

Figure 10 shows the surface pressure distribution as a function of the longitudinal position for the wind side. The computed result is compared with experimental data[14] (circles) and is seen to be in very good agreement. The expansions and recompressions near the ogive–cylinder and cylinder–boat-tail junctions are captured adequately by the computations. Comparison of the surface pressure on the lee side is shown in Figure 11. A small discrepancy can be seen on the cylinder and boat-tail sections. The computed result, however, is generally in good agreement

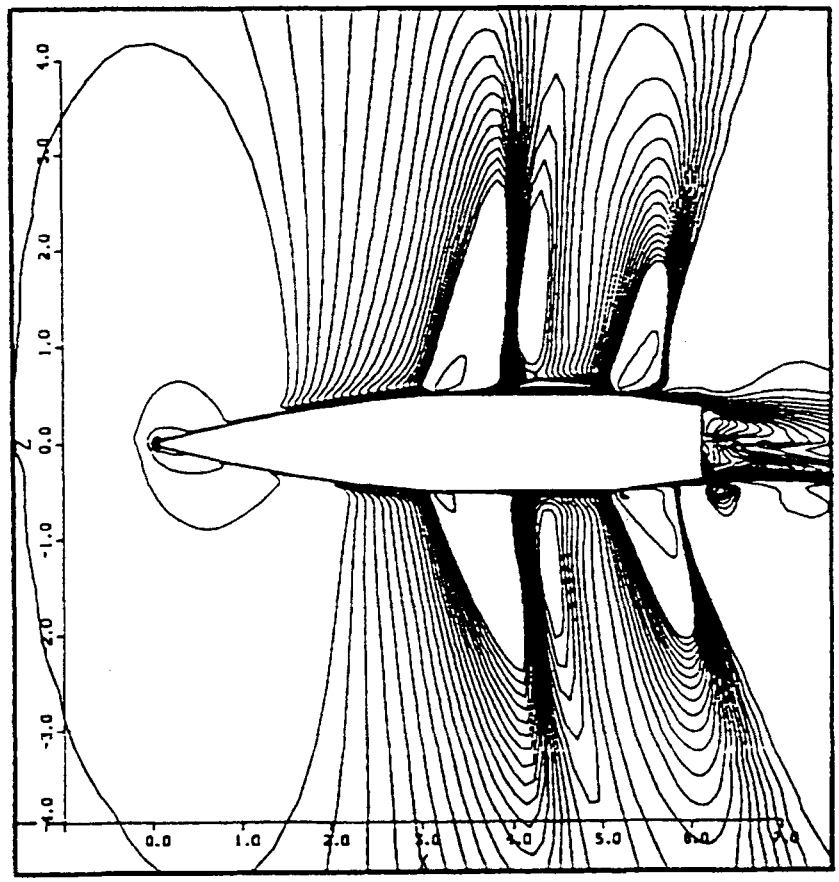

Figure 8. Mach contours; $M_\infty = 0{\cdot}96$, $\alpha = 4°$

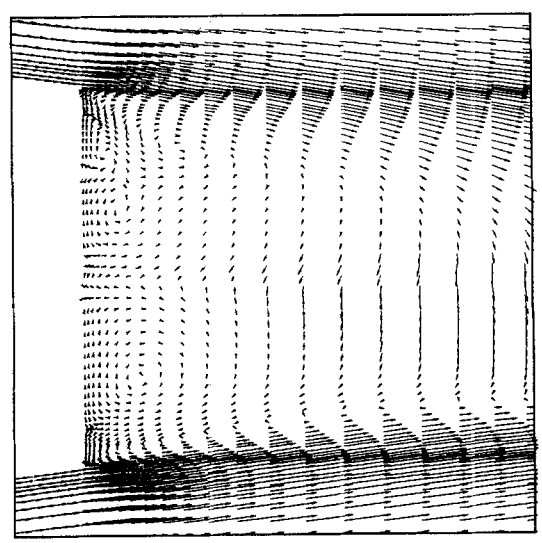

Figure 9. Velocity vectors in the base region; $M_\infty = 0{\cdot}96$, $\alpha = 4°$

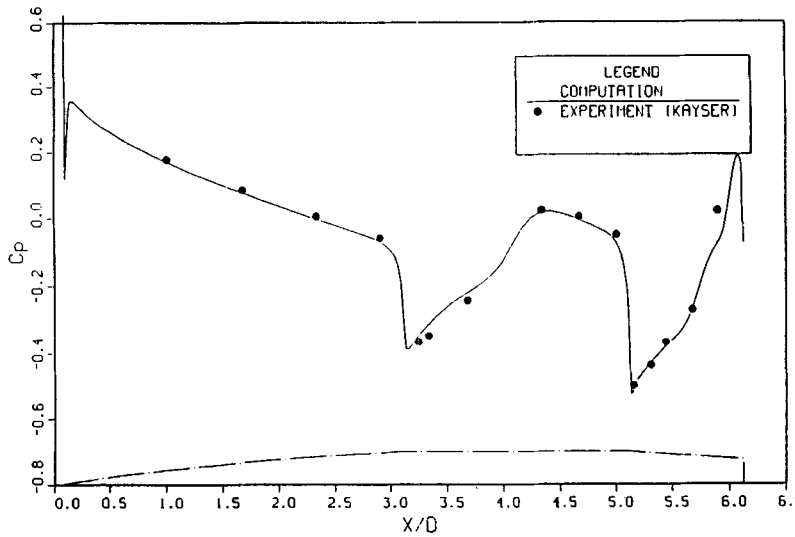

Figure 10. Longitudinal surface pressure distribution, wind side; $M_\infty = 0{\cdot}96$, $\alpha = 4°$

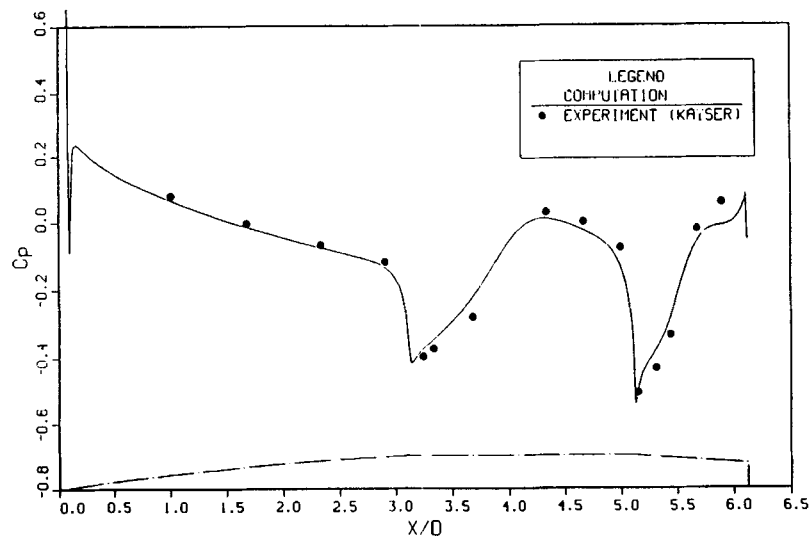

Figure 11. Longitudinal surface pressure distribution, lee side; $M_\infty = 0{\cdot}96$, $\alpha = 4°$

with the experimental data. Figures 12 and 13 show the circumferential surface pressure distributions for longitudinal positions $X/D = 5{\cdot}19$ and $5{\cdot}56$ respectively. Both of these longitudinal positions are on the boat-tail. In these figures the present computed result is compared with experimental data (circles) and also with a previous computational result obtained on a rather coarse mesh.[3] Apart from the small discrepancy near the lee side ($\phi = 180°$), the present computed result is in good agreement with the experimental data and is more accurate than the previous computed result (see Figure 12). The agreement of the present result with experiment is

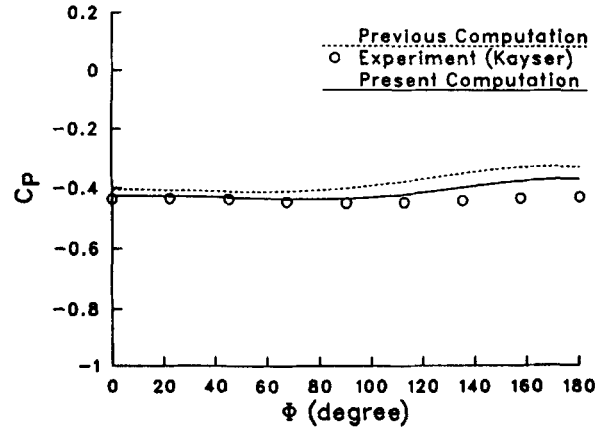

Figure 12. Circumferential surface pressure distribution; $M_\infty = 0{\cdot}96$, $\alpha = 4°$, $X/D = 5{\cdot}19$

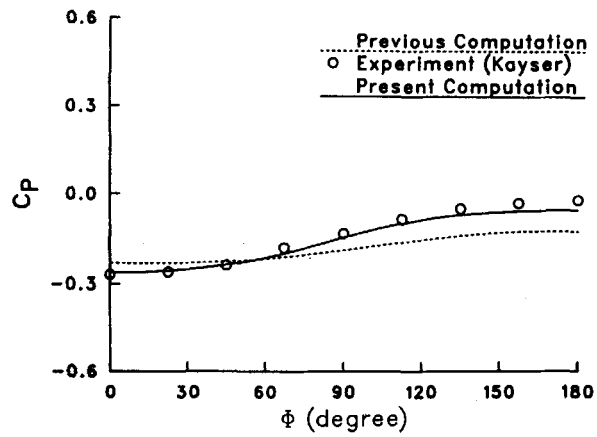

Figure 13. Circumferential surface pressure distribution, $M_\infty = 0{\cdot}96$, $\alpha = 4°$, $X/D = 5{\cdot}56$

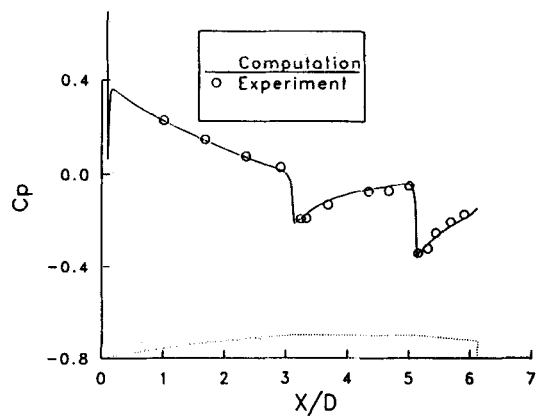

Figure 14. Longitudinal surface pressure distribution, wind side; $M_\infty = 1{\cdot}1$, $\alpha = 4°$

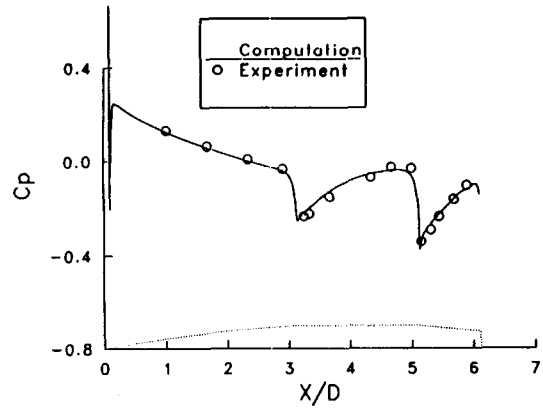

Figure 15. Longitudinal surface pressure distribution, lee side; $M_\infty = 1{\cdot}1$, $\alpha = 4°$

very good near the wind side ($\phi = 0$). Figure 13 shows the circumferential surface pressure comparison at a longitudinal position $X/D = 5{\cdot}56$. As seen in this figure, the present numerical result is in excellent agreement with the experimental data. The improved accuracy of the present numerical prediction compared with the previous coarse grid solution is obvious.

Numerical solution has also been obtained at a high transonic speed $M_\infty = 1.1$ and $\alpha = 4°$. Comparisons of the longitudinal surface pressure distributions for both wind side and lee side have been made with experimental data[14] and are shown in Figures 14 and 15. The agreement of the computed surface pressures with experiment is very good. At this high transonic Mach number, the shocks on the cylinder as well as on the boat-tail are very weak as evidenced by the absence of a sharp rise in pressure in those areas. Strong shock waves occur at lower transonic Mach numbers, for example at $M_\infty = 0{\cdot}96$ and $\alpha = 4°$ (see Figure 10). The expansions and recompressions near the ogive–cylinder and cylinder–boat-tail junctions can be clearly observed in Figures 14 and 15.

4. CONCLUDING REMARKS

In conjunction with a new Navier–Stokes code, a simple composite grid scheme has been developed which allows the fine computational grids needed for accurate tansonic flow computations to be obtained on a CRAY X-MP/48 computer. The numerical method uses an implicit, approximately factored, partially upwind (flux-split) algorithm.

The accuracy and stability of the composite grid scheme have been tested by numerically simulating the flow over an ellipsoid at angle of attack ($M = 0{\cdot}90$, $\alpha = 10°$) and comparing this solution with a single-grid solution. The flow fields over a projectile at $M_\infty = 0{\cdot}96$, $1{\cdot}1$ and $\alpha = 4°$ have ben computed using a fine grid and compared to experiment. The computed surface pressures are in very good agreement with the experimental data. Currently work is in progress to analyse and determine the accuracy of the wake region flow. Future work will be directed to determining the critical aerodynamic behaviour at transonic speeds.

ACKNOWLEDGEMENT

This work was performed during J. Sahu's visit to NASA Ames Research Center.

REFERENCES

1. G. S. Deiwert, 'Numerical simulation of three dimensional boattail afterbody flowfield', *AIAA J.*, **19,** 582–588 (1981).
2. C. J. Nietubicz *et al.*, 'Computations of projectiles magnus effect at transonic velocities', *AIAA J.*, **23,** 998–1004 (1985).
3. J. Sahu, 'Three dimensional base flow calculation for a projectile at transonic velocity', *AIAA Paper No. 86–1051*, May 1986.
4. R. Beam and R. F. Warming, 'An implicit factored scheme for the compressible Navier–Stokes equations', *AIAA Paper No. 85–1815-CP*, August 1985.
5. J. L. Steger, 'Implicit finite difference simulation of flow about arbitrary geometries with application to airfoils', *AIAA J.*, **16,** 679–686 (1978).
6. T. H. Pulliam and J. L. Steger, 'On impilicit finite-difference simulations of three-dimensional flow', *AIAA J.*, **18,** 159–167 (1980).
7. S. X. Ying, J. L. Steger, L. B. Schiff and D. Baganoff, 'Numerical simulation of unsteady, viscous, high-angle-of-attack flows using a partially flux-split algorithm', *AIAA Paper No. 86–2179*, August 1986.
8. J. L. Steger and R. F. Warming, 'Flux vector splitting of the inviscid gasdynamic equations with application to finite-difference methods', *J. Comput. Phys.*, **40,** 263–293 (1981).
9. J. L. Steger and P. Buning, 'Developments in the simulation of compressible inviscid and viscous flow on supercomputers', *NASA TM 86674*, 1985.
10. J. A. Benek *et al.*, 'A 3-D Chimera grid embedding technique', *AIAA Paper No. 85–1523*, July 1985.
11. G. R. Srinivasan, W. J. Chyu and J. L. Steger, 'Computation of simple three-dimensional wing–vortex interaction in transonic flow', *AIAA Paper No. 81–1206*, 1981.
12. T. H. Pulliam and J. L. Steger, 'Recent improvements in efficiency, accuracy and convergence of implicit approximate factorization algorithms', *AIAA Paper No. 85–0360*, January 1985.
13. B. S. Baldwin and H. Lomax, 'Thin layer approximation and algebraic model for separated turbulent flows. *AIAA Paper No. 78–257*, 1978.
14. L. D. Kayser, F. Whiton, 'Surface pressure measurements on a boattailed projectile shape at transonic speeds', *U.S. Army Ballistic Research Laboratory, ARBRL-MR-03161*, March 1982.
15. L. D. Kayser, 'Base pressure measurements on a projectile shape at Mach numbers from 0·91 to 1·20', *U.S. Army Ballistic Research Laboratory, ARBRL-MR-03353*, April 1984.