AIAA 2001-0438
Implicit Parallel Algorithms for Time-accurate Simulations of Hypersonic Boundary Layer Receptivity over Blunt Bodies

Haibo Dong and Xiaolin Zhong
University of California, Los Angeles
Los Angeles, CA

39th Aerospace Sciences Meeting & Exhibit
January 8–11, 2001 / Reno, NV

Implicit Parallel Algorithms for Time-accurate Simulations of Hypersonic Boundary Layer Receptivity over Blunt Bodies

Haibo Dong *and Xiaolin Zhong †
University of California, Los Angeles, California 90095

Abstract

Due to the progress in computer techniques in recent years, distributed memory parallel systems are rapidly gaining importance in direct numerical simulation of the stability and transition of compressible boundary layers. However, most of the numerical methods used are explicit high-order finite-difference methods. The difficulty in using explicit methods for DNS of hypersonic flow is the limitation of the temporal steps since the Navier-Stokes equations are stiff for explicit numerical schemes. A short marching step can not be avoided during the computation due to stability limitation. This requires implicit treatment in algorithms. In this paper, we extend our previous work to develop and validate efficient and high-order time accurate parallel implicit numerical methods including parallel Fourier spectral methods on massively distributed memory computers to solve unsteady Navier-Stokes equations. Divide and conquer method and parallel iterative method are considered as the block linear system solver. The efficiency and accuracy of the new parallel implicit code has been tested in solving a 2-D convection-diffusion model equation first. Several test cases, which include the numerical simulations of the supersonic flow over a sphere, and hypersonic boundary layer receptivity to freestream acoustic waves over 3-D blunt wedge, are used to evaluated the efficiency and accuracy of the new parallelized code. At last, on-going work of simulating nonlinear wave breakdown process of instability waves in hypersonic boundary layers over a 3-D parabolic leading edge is studied by implementing the new algorithms.

1 Introduction

The prediction of laminar-turbulent transition in hypersonic boundary layers is a critical part of the aerodynamic design and control of hypersonic vehicles [1]. Due to the progress in computer techniques in recent years, distributed memory parallel systems are rapidly gaining importance in fields where computational performance is important such as computational fluid dynamics (CFD). On massively parallel systems, domain decomposition method is used to distribute the load between processors by dividing the computational domain into sub-domains. This makes direct numerical simulation become a powerful tool in the study of fundamental flow physics of the stability and transition of boundary layers [2,3]. There has been considerable research interest in parallel algorithms of solving block matrices on massively distributed memory computers and direct numerical simulations of laminar-turbulent transition of compressible boundary layers.

In DNS studies, the full unsteady Navier-Stokes equations are numerically simulated without using any empirical turbulence models. The development of the instability waves and the nonlinear breakdown processes are numerically captured by the simulations. Though such simulations are computationally intensive, they have the ability to simulate many of the physical effects that are not possible by other approaches. Most of the numerical methods used are explicit high-order finite-difference methods in the non-periodic streamwise and wall-normal directions, and the Fourier spectral methods in the periodic spanwise direction [4–6], assuming a simple flat surface without the presence of shock waves. But it is difficult to apply existing numerical methods for compressible boundary layer DNS to hypersonic boundary layers over blunt bodies. For viscous flow calculations, the extremely small grid sizes in the boundary layers near the wall are used. The stiffness of the governing equations refers to the fact that the time steps required by the stability requirement in the calculations are much smaller than that needed by accuracy consideration so that it is difficult to perform the simulation in reasonable computation time. This requires implicit treatment in algorithms.

Generally, works of implementing implicit schemes on parallel computers are based on the approximate factorization method employing either an alternating direction implicit(ADI) [7] or a lower-upper(LU) factorization [8]. However, these works mainly focus on simulation of 2-D flow and at most second-order-accurate time stepping is used. There have been very few direct numerical simulation of hypersonic boundary layer stability and transition on complicated 3-D configurations by using high-order time-accurate implicit algorithms on parallel computers. Implicit schemes are more difficult to be parallelized primarily because of the

*Graduate Student, Mechanical and Aerospace Engineering Department, Student Member AIAA, haibo@seas.ucla.edu
†Associate Professor, Mechanical and Aerospace Engineering Department, Associate Fellow AIAA, xiaolin@seas.ucla.edu.

inherent global spatial dependencies for the solution of large systems in the form of block tridiagonal or more diagonal matrices. Thus the existing algorithms for sequential computers to distributed memory computers usually requires some algorithmic changes.

Many mathematicians have paid much attention on how to solve the linear systems especially block linear systems which arise for example in the numerical solution of partial differential equations on parallel computers. Parallel algorithms have been developed from fine-grained forms of parallelism to coarse-grained forms of parallelism. For instance, parallel iterative methods [9,10] which include iterative Jacobian method, preconditioned conjugate gradient method [11] and divide and conquer method [12,13] are commonly used to solve the block matrices on parallel computers. On the other hand, Fourier spectral methods have been applied to solve the problem with periodic boundary conditions on parallel computers [14,15] because of their high accuracy. Methods of mono-dimensional and multi-dimensional parallel FFT algorithms [16], parallel algorithms of the Fourier pseudospectral method [15], parallel spectral method with the Local Fourier Basis [17] are the typical works in this field.

Recently, [18–20] has developed and validated a set of fifth and seventh-order shock-fitting schemes for the DNS of practical 2-D and 3-D hypersonic flows over planar or axisymmetric blunt bodies. Recently, we [21,22] we have already implemented parallel method to our explicit DNS code and semi-implicit DNS code to study the hypersonic boundary-layer receptivity to freestream disturbances over an elliptic cross-section cone and 3-D parabola wedge on IBM SP2 computers. In semi-implicit scheme, non-stiff terms in the governing equations are treated explicitly and simultaneously the stiff terms are treated implicitly to overcome the stiffness of the Navier-Stokes equations.

The objective of this paper is to extend the previous 3-D parallel high-order schemes with shock-fitting method in [20,22] to solve compressible Navier-Stokes equations implicitly. Divide and conquer algorithm and parallel iterative methods are used to solve the big banded Jacobian matrices from the implicit methods. Meanwhile, parallel Fourier spectral method is applied to the periodic spanwise direction. A 2-D convection-diffusion model equation has been used to test the accuracy and efficiency of the new parallelized implicit methods. Analytical comparisons have been made to study the efficiency of different parallelized algorithms in solving the model equation. DNS of the supersonic flow over a sphere and hypersonic boundary layer receptivity over 3-D blunt wedge are used to investigate the numerical accuracy and efficiency of the new parallelized code with shock-fitting method. Accuracy and efficiency are investigated in these test cases. At last, on-going work of simulating nonlinear wave breakdown process of instability waves in hypersonic boundary layers over a parabolic leading edge is presented by implementing the new algorithms.

2 Governing Equations and Numerical Methods

2.1 Governing Equations

The governing equations are the unsteady three-dimensional Navier-Stokes equations written in a conservation-law form

$$\frac{\partial U}{\partial t}+\frac{\partial F_j}{\partial x_j}+\frac{\partial F_{vj}}{\partial x_j}=0 \tag{1}$$

where

$$U = \{\rho,\ \rho u_1,\ \rho u_2,\ \rho u_3,\ e\} \tag{2}$$

$$F_j = \begin{Bmatrix} \rho u_j \\ \rho u_1 u_j + p\delta_{1j} \\ \rho u_2 u_j + p\delta_{2j} \\ \rho u_3 u_j + p\delta_{3j} \\ (e+p)\,u_j \end{Bmatrix} \tag{3}$$

$$F_{vj} = \begin{Bmatrix} 0 \\ \tau_{1j} \\ \tau_{2j} \\ \tau_{3j} \\ \tau_{jk}u_k - q_j \end{Bmatrix} \tag{4}$$

$$p = \rho RT \tag{5}$$

$$e = \rho(c_v T + \frac{\rho}{2}u_k u_k) \tag{6}$$

$$\tau_{ij} = -\mu\left(\frac{\partial u_i}{\partial x_j}+\frac{\partial u_j}{\partial x_i}\right)+2\mu/3\frac{\partial u_k}{\partial x_k}\delta_{ij} \tag{7}$$

$$q_j = -\kappa\frac{\partial T}{\partial x_j} \tag{8}$$

The details for the expressions above can be found in [18]. The viscosity and heat conductivity coefficients are computed by the Sutherland law and the assumption of a constant Prandtl number. Perfect gas assumption is used in all flows considered in this paper, though the method presented here can be easily extended to nonequilibrium real-gas hypersonic flows.

For numerical simulations of flow fields over a curved body surface, structured body fitted grids are used to transform the governing equations (1) in the Cartesian coordinates into a set of curvilinear three-dimensional coordinates (ξ, η, ζ, τ) along the body fitted grid lines. The transformation relations for the two set of coordinates are

$$\begin{cases} \xi = \xi(x,y,z) \\ \eta = \eta(x,y,z,t) \\ \zeta = \zeta(x,y,z) \\ \tau = t \end{cases} \Longleftrightarrow \begin{cases} x = x(\xi,\eta,\zeta,\tau) \\ y = y(\xi,\eta,\zeta,\tau) \\ y = y(\xi,\eta,\zeta,\tau) \\ t = \tau \end{cases} \tag{9}$$

The governing equations (1) are transformed into the computational domain (ξ, η, ζ, τ) as follows

$$\frac{1}{J}\frac{\partial U}{\partial \tau}+\frac{\partial E'}{\partial \xi}+\frac{\partial F'}{\partial \eta}+\frac{\partial G'}{\partial \zeta}$$

$$+\frac{\partial E'_v}{\partial \xi}+\frac{\partial F'_v}{\partial \eta}+\frac{\partial G'_v}{\partial \zeta}+U\frac{\partial(\frac{1}{J})}{\partial \tau}=0 \qquad (10)$$

where

$$E' = \frac{F_1\xi_x+F_2\xi_y+F_3\xi_z}{J} \qquad (11)$$

$$F' = \frac{F_1\eta_x+F_2\eta_y+F_3\eta_z+U\eta_t}{J} \qquad (12)$$

$$G' = \frac{F_1\zeta_x+F_2\zeta_y+F_3\zeta_z}{J} \qquad (13)$$

$$E'_v = \frac{F_{v1}\xi_x+F_{v2}\xi_y+F_{v3}\xi_z}{J} \qquad (14)$$

$$F'_v = \frac{F_{v1}\eta_x+F_{v2}\eta_y+F_{v3}\eta_z}{J} \qquad (15)$$

$$G'_v = \frac{F_{v1}\zeta_x+F_{v2}\zeta_y+F_{v3}\zeta_z}{J} \qquad (16)$$

where J is the Jacobian of the coordinate transformation, and ξ_x, ξ_y, ξ_z, η_x, η_y, η_z, η_t, ζ_x, ζ_y, and ζ_z are the grid transformation matrices. In the equations, the transformed inviscid fluxes E', F', and G' are standard flux terms with known eigenvalues and eigenvectors. The transport flux terms E'_v, F'_v, and G'_v contain both first-order and second-order spatial derivatives of velocity and temperature. These derivatives in the Cartesian coordinates (x, y, z) are transformed into the computational coordinates (ξ, η, ζ) using a chain rule for spatial discretization.

2.2 Implicit Method

2.2.1 High-Order Semi-Implicit Method

The stiffness of viscous flow simulations is mainly due to terms associated with derivatives in the wall-normal direction ($\frac{\partial()}{\partial\eta}$ and $\frac{\partial^2()}{\partial\eta^2}$) because of grid stretching near the wall. Therefore, Navier-Stocks Equation (10) for a three-dimensional flow in (ξ, η, ζ, τ) is additively split into relatively nonstiff part $\mathbf{f}(U_{ijk})$ and stiff part $\mathbf{g}(U_{ijk})$ as follows

$$\frac{1}{J}\frac{\partial U_{ijk}}{\partial t} = \mathbf{f}(U_{ijk})+\mathbf{g}(U_{ijk}) \qquad (17)$$

where details can be found in Ref. [22]. In Eq. (17), $\mathbf{g}(U_{ijk})$ is much stiffer than $\mathbf{f}(U_{ijk})$ since grid spacing in the wall-normal direction is much smaller than that used in streamwise direction for most viscous flow simulations. Therefore, high-order semi-implicit method can be used to overcome the stiffness of $\mathbf{g}(U_{ijk})$ while maintaining high-order temporal accuracy.

The split governing equation (17) is first approximated by high-order accurate finite difference methods. For the case of direct numerical simulation of compressible boundary layers with a bow shock, the shock wave can be treated by a shock-fitting method because there is no discontinuity in the interior of the computational domain. In this paper, a fifth-order upwind scheme [20] is used to discretize the inviscid flux derivatives. Meanwhile, high-order central difference schemes, such as the sixth-order central scheme, are used to discretize the viscous flux terms in the equations. A simple local Lax-Friedrichs scheme is used to split the inviscid flux vectors into positive and negative wave fields. Details of derivations can be found in Ref. [22]. The spatial discretization of the split Eq. (17) using these high-order schemes coupled with appropriate boundary conditions leads to a system of ordinary differential equations in the form of

$$\frac{d\mathbf{u}}{dt} = [\mathbf{f}(t,\mathbf{u})]+[\mathbf{g}(t,\mathbf{u})] \qquad (18)$$

where $\mathbf{u} = \{U_{ij}, \text{for } i = 1,\cdots IL,\ j = 1,\cdots JL\}$ is the vector of all discretized variables in the flow field, $[\mathbf{f}(t,\mathbf{u})]$ represents the discretized nonstiff term, and $[\mathbf{g}(t,\mathbf{u})]$ represents the discretized stiff term.

The system of ordinary differential equations of Eq. (17) can be integrated in time using semi-implicit temporal schemes, where $\mathbf{f}$ is treated explicitly and $\mathbf{g}$ is treated implicitly. It was shown by Zhong [23] that in order to have a third or higher order temporal accuracy, the semi-implicit method need to be derived in a way that the effects of coupling between the implicit and explicit terms on the accuracy need be considered. Zhong [23] subsequently derived three kinds of third-order semi-implicit Runge-Kutta schemes for high-order temporal integration of the governing equations for reacting flow simulations. High-order low-storage semi-implicit Runge-Kutta method versions(LSSIRK) have also been derived in [24]. In particular, Rosenbrock type Runge-Kutta (LSSIRK-rC) Method can be written as following

$$\begin{cases} [\mathbf{I}-hc_j\mathbf{J}(\mathbf{u}_{j-1}+\bar{c}_j\mathbf{k}_{j-1})]\,\mathbf{k}_j = \\ \qquad h\,(\mathbf{f}(\mathbf{u}_{j-1})+\mathbf{g}(\mathbf{u}_{j-1}+\bar{c}_j\mathbf{k}_{j-1})) \\ \qquad +a_j\,[\mathbf{I}-hc_j\mathbf{J}(\mathbf{u}_{j-1}+\bar{c}_j\mathbf{k}_{j-1})]\,\mathbf{k}_{j-1} \\ \mathbf{u}_j = \mathbf{u}_{j-1}+b_j\mathbf{k}_j \end{cases} \qquad (19)$$

where $j = 1, ..., r$ and parameters c_j, $\bar{c}_j$, b_j can be found in [24]. For instance, in LSSIRK-3C, $b_1 = \frac{1}{4}$, $b_2 = \frac{2}{9}$, $b_3 = 3$, $a_1 = -\frac{1}{4}$, $a_2 = -\frac{29}{27}$, $c_1 = 2.26760$, $c_2 = 2.68530$, $c_3 = 2.30975$, $\bar{c}_2 = -1.14310$, and $\bar{c}_3 = -2.03122$. The parameters of the semi-implicit Runge-Kutta methods are chosen by both stability and accuracy requirements with the simultaneous coupling between the explicit and implicit terms.

In applying the semi-implicit method to Navier-Stokes equation (18), global Jacobian matrices J comes from the implicit method and can be defined by $\mathbf{J}(\mathbf{u}) = \partial\mathbf{g}/\partial\mathbf{u}$. As the derivations in Ref. [25], the fifth-order upwind scheme and sixth-order central scheme are used to approximate the derivatives of $\frac{\partial()}{\partial\eta}$ and $\frac{\partial^2()}{\partial\eta^2}$. This leads to

$$\begin{aligned}\delta\mathbf{g}(U_{ijk}) = {} & \mathbf{A}_{ijk}\delta U_{ij-3k}+\mathbf{B}_{ijk}\delta U_{ij-2k}+\mathbf{C}_{ijk}\delta U_{ij-1k} \\ & +\mathbf{D}_{ijk}\delta U_{ijk}+\mathbf{E}_{ijk}\delta U_{ij+1k}+\mathbf{F}_{ijk}\delta U_{ij+2k} \\ & +\mathbf{G}_{ijk}\delta U_{ij+3k} \end{aligned} \qquad (20)$$

where the coefficient matrices can also be found in Ref. [25].

The final global Jacobian matrix for the system of ordinary differential equations, Eq. (18), is a block seven-diagonal matrix involving terms along the j grid direction only. This block seven-diagonal system of equations can be solved efficiently by a banded matrix solver.

The physical boundary conditions for viscous flows are non-slip condition for velocity and isothermal or adiabatic condition for temperature. The freestream flow conditions are specified by a given flow. For the flow disturbed by disturbances, the disturbances are specified according to the particular physical nature of the disturbances. Since the emphasis of current paper is the parallelized semi-implicit method for efficient and accurate time integration of the governing equations, we will mainly consider flows with a supersonic exit where the reflection of disturbances are negligible.

Boundary conditions on the wall are included in the global Jacobian matrix to ensure that it is a global implicit equation and advanced in time. For example, to include the lower wall boundary conditions, by imposing the fourth-order boundary conditions, δU_{i1} can be written as,

$$\delta U_{i1} = (\frac{\partial U}{\partial V})_{i1}(\frac{\partial V_{i1}}{\partial V_{i2}})(\frac{\partial V}{\partial U})_{i2}\delta U_{i2} + (\frac{\partial U}{\partial V})_{i1}(\frac{\partial V_{i1}}{\partial V_{i3}})(\frac{\partial V}{\partial U})_{i3}\delta U_{i3} + (\frac{\partial U}{\partial V})_{i1}(\frac{\partial V_{i4}}{\partial V_{i4}})(\frac{\partial V}{\partial U})_{i4}\delta U_{i4} + (\frac{\partial U}{\partial V})_{i5}(\frac{\partial V_{i1}}{\partial V_{i5}})(\frac{\partial V}{\partial U})_{i5}\delta U_{i5}$$

where V and $\frac{\partial V}{\partial U}$ can be found in appendix, $\frac{\partial V_{i1}}{\partial V_{ij}}(j = 2, ..., 5)$ is the coefficients of the corresponding boundary conditions applied to the lower wall.

2.2.2 Two-sweep Gauss-Seidel Line Relaxation Method

To run the code implicitly, two-sweep Gauss-Seidel line relaxation method is considered as one way to treat both streamwise and wall-normal direction implicitly. Equation (10) for a two-dimensional flow in (ξ, η, τ) is written as follow

$$\frac{1}{J}\frac{\partial U_{ij}}{\partial \tau} + \left(\frac{\partial E'}{\partial \xi} + \frac{\partial E'_v}{\partial \xi} + \frac{\partial F'}{\partial \eta} + \frac{\partial F'_v}{\partial \eta} + U\frac{\partial(\frac{1}{J})}{\partial \tau}\right)_{ij} = 0 \quad (21)$$

where i and j are the grid indices in the ξ and η direction respectively.

The implicit formulation of above equation is

$$\delta(U/J)_{ij}^{n+1} + \Delta\tau\left(\delta E'^{n+1} + \delta E'^{n+1}_v + \delta F'^{n+1} + \delta F'^{n+1}_v\right)_{ij} = \Delta U_{ij}^n \quad (22)$$

Two-sweep Gauss-Seidel line relaxation method is applied to solve Eq. (21).

Backward sweep:

$$\mathbf{A}\delta U^*_{ij} + \mathbf{B}\delta U^*_{ij-1} + \mathbf{C}\delta U^*_{ij-2} + \mathbf{D}\delta U^*_{ij+1} + \mathbf{E}\delta U^*_{ij+2} = \Delta U_{ij} + \mathbf{D_1}\delta U^*_{i+1j} + \mathbf{E_1}\delta U^*_{i+2j} \quad (23)$$

Forward sweep:

$$\mathbf{A}\delta U^{n+1}_{ij} + \mathbf{B}\delta U^{n+1}_{ij-1} + \mathbf{C}\delta U^{n+1}_{ij-2} + \mathbf{D}\delta U^{n+1}_{ij+1} + \mathbf{E}\delta U^{n+1}_{ij+2} = \Delta U_{ij} + \mathbf{B_1}\delta U^{n+1}_{i-1j} + \mathbf{C_1}\delta U^{n+1}_{i-2j} + \mathbf{D_1}\delta U^{n+1}_{i+1j} + \mathbf{E_1}\delta U^{n+1}_{i+2j} \quad (24)$$

where the coefficient matrices are relative the derivatives of u, v, p, T. For each sweep, the left side is solved by good pentadiagonal solver as described in following.

2.3 Parallel Methods of Solving Block Linear Systems

The standard (portable) message passing interface(MPI) is the parallel library we used to parallelize our code. A cluster of workstations are used to run our parallel codes. The present configuration has 24 RISC/6000 processors. The main memory capacity of these nodes is 256 megabytes. Divide and Conquer (DAC) method and parallel iterative methods are presented to solve the block linear systems on massively distributed memory computers. These methods have been extended to solve the big Jacobian matrices of implicit algorithms during numerically solving high-order discretized nonlinear Navier-Stokes equations.

2.3.1 Divide and Conquer Method

Though the divide and conquer method is developed for solving the multi-diagonal matrix on MIMD parallel computers, only tridiagonal matrix solver using 3 nodes are presented here for simplicity. The extension to more diagonal matrix solver using more nodes is straightforward.

Consider the tridiagonal linear system $Tx = d$ with $n = pk$ equations. The divide and conquer method divides the given tridiagonal system into p parallel tasks of size $k = n/p$. T is partitioned into block tridiagonal form with each diagonal block a $k \times k$ tridiagonal matrix and each subdiagonal block a $k \times k$ null matrix, except for one single nonzero element on its upper right (lower left) corner. The tridiagonal system can be written in block form as

$$\begin{pmatrix} T_1 & C_1 & & \\ B_1 & T_2 & \ddots & \\ & \ddots & \ddots & C_{p-1} \\ & & B_p & T_p \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_p \end{pmatrix} = \begin{pmatrix} d_1 \\ d_2 \\ \vdots \\ d_p \end{pmatrix}$$

where T_j are tridiagonal matrices,

$$T_j = \begin{pmatrix} a_{(j-1)k+1} & c_{(j-1)k+1} & & \\ b_{(j-1)k+2} & a_{(j-1)k+2} & \ddots & \\ & \ddots & \ddots & c_{jk-1} \\ & & b_{jk} & a_{jk} \end{pmatrix}$$

and

$$B_j = \begin{pmatrix} 0 & \cdots & b_{(j-1)k+1} \\ & \ddots & \\ & & \\ 0 & \cdots & 0 \end{pmatrix}$$

$$C_j = \begin{pmatrix} 0 & \cdots & 0 \\ & \ddots & \\ & & \\ c_{(j-1)k} & \cdots & 0 \end{pmatrix}$$

where $j = 1, 2, ..., p$. $\mathbf{x}$ and $\mathbf{d}$ are partitioned to conform with the blocks of T:

$$x_j = \begin{pmatrix} x_{(j-1)k+1} \\ x_{(j-1)k+1} \\ \vdots \\ x_{(j-1)k+p} \end{pmatrix} d_j = \begin{pmatrix} d_{(j-1)k+1} \\ d_{(j-1)k+2} \\ \vdots \\ d_{(j-1)k+p} \end{pmatrix}$$

For $j = 1, 2, ..., p$, by defining

$$\begin{aligned} y_j &= {T_j}^{-1} d_j \\ z_1 &= {T_1}^{-1} e_k \\ z_m &= {T_j}^{-1} e_1 \qquad\qquad z_{m+1} = {T_j}^{-1} e_k \\ z_{2p-2} &= {T_p}^{-1} e_1 \end{aligned}$$

where $m = 2i - 2$, e_1 and e_k are the first and last unit vector. We have the following implicit relations

$$\begin{aligned} x_1 &= y_1 + \alpha_1 z_1 \\ x_j &= y_j + \alpha_{2j-2} z_{2j-2} + \alpha_{2j-1} z_{2j-1} \\ & \qquad 2 \le j \le p-1 \\ x_p &= y_p + \alpha_{2p-2} z_{2p-2} \end{aligned}$$

where α_1, α_2,..., α_{2p-2}, α_4 satisfy the relations as following reduced systems:

$$\begin{pmatrix} {e_k}^T z_1 & 1/b_{k+1} & & \\ 1/c_k & {e_1}^T z_2 & \ddots & \\ & \ddots & \ddots & 1/b_{k(p-1)+1} \\ & & 1/c_{k(p-1)} & {e_1}^T z_{2p-2} \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_{2p-2} \end{pmatrix} = \begin{pmatrix} {e_k}^T y_1 \\ {e_1}^T y_2 \\ \vdots \\ {e_1}^T y_{2p-2} \end{pmatrix}$$

From the solution of above equations, we then obtain the desired solution x_1, x_2, x_3. Two kinds of approaches, all-to-all broadcast [13] and folded skip-decoupling method [12], are used to solve above reduced systems.

2.3.2 Parallel Iterative Method

Parallel iterative methods are also used to solve the banded matrix and dense matrix on MIMD machines. In each iteration, the coefficient matrix is used to perform matrix-vector and matrix-matrix multiplications. The number of iterations required to solve a system of equations with a desired precision is usually data dependent; hence, the number iterations is not known prior to executing the algorithm. If iterative methods yield a solution, they are usually less expensive than direct methods for matrix factorization, especially for solving dense matrix. Parallel Jacobi iterative method and the parallel preconditioned conjugate gradient iterative method are tested in our work.

Parallel Jacobi Iterative Method

The i^{th} equation of a system of linear equations $Tx = d$ is:

$$\sum_{j=0}^{n-1} T_{ij} x_j = d_j$$

If all the diagonal elements of T are nonzero, a typical iteration step of the above equation can be written as:

$$\begin{aligned} {x_i}^k &= \frac{1}{T_{ij}} (d_i - \sum_{j \neq i} T_{ij} {x_j}^{k-1}) \\ &= \frac{1}{T_{ij}} (d_i - \sum_{j=0}^{n-1} T_{ij} {x_j}^{k-1}) + {x_i}^{k-1} \\ &= \frac{{r_i}^{k-1}}{T_{ii}} + {x_i}^{k-1} \end{aligned}$$

where ${r_i}^{k-1} = d_i - \sum_{j=0}^{n-1} T_{ij} {x_j}^{k-1}$ is the residual after k iterations.

Each iteration of the Jacobi method given in above performs three main computations: the vector inner product, the calculation of ${x_i}^k$, the matrix-vector multiplication. After proper mapping the matrix to different processors, calculation of ${x_i}^k$ can be done in parallel without any communication, vector inner product and sparse matrix-vector multiplication can be executed in parallel with communication.

Parallel Preconditioned Conjugate Gradient Method

The conjugate gradient method belongs to a class of iterative methods known as minimization methods. If the coefficient matrix T has l distinct eigen values, the conjugate gradient algorithm converges to the solution of the system $Tx = d$ in at most l iterations. Therefore, if T has many distinct eigenvalues that vary widely in magnitude, the conjugate gradient algorithm may require a large number of iterations to converge to an acceptable approximation to the solution. The speed of convergence of the conjugate gradient algorithm can

be increased by preconditioning T with the congruence transformation $\hat{T} = RTR^T$, where R is a nonsingular matrix. R is chosen such that $\hat{T}$ has fewer distinct eigenvalues than T.

An typical iteration of preconditioned conjugate gradient method is:

$$x_k = x_{k-1} + \alpha_k p_k$$

where

$$\begin{aligned} \alpha_k &= \frac{\gamma_{k-1}}{{p_k}^T T p_k} \\ \gamma_k &= {r_k}^T z_k \\ p_{k+1} &= z_k + \gamma_k p_k / \gamma_{k-1} \\ r_k &= r_{k-1} - \alpha_k T p_k \end{aligned}$$

where z_k is calculated from $Mz_k = r_k$. $M = (R^T R)^{-1}$ is the preconditioner matrix. The parallel preconditioned conjugate gradient algorithm involves the following four types of computations in each iteration: matrix plus matrix, vector inner products, matrix-vector multiplication, and solving the system $Mz_k = r_k$.

2.4 Parallel Fourier Spectral Method

As the basic part of the parallel Fourier spectral method, discrete Fourier transform (DFT) has been studied by many researchers. In 1965, Cooley and Tukey devised an algorithm to compute the DFT of an n-point series in $O(n \log n)$ operations. Their new algorithm was a significant improvement over previously known methods for computing the DFT, which required $O(n^2)$ operations. The revolutionary algorithm by Cooley and Tukey and its variations are referred to as the fast Fourier transform (FFT). Due to its wide application in scientific and engineering fields, there has been a lot of interest in implementing FFT on parallel computers.

In our research, parallel Fourier spectral method is used to solve the derivatives in periodic spanwise direction on massively distributed memory parallel computers. The binary-exchange algorithm which is based on iterative FFT algorithm is implemented to compute an n-point FFT on p processors, when $n > p$. Assuming that both n and p are powers of two with $p = 2^d$ and $n = 2^r$, the sequences are partitioned into blocks of n/p contiguous elements and assign one block to each processor. If $(b_0 b_1 ... b_{r-1})$ is the binary representation of any i, then $R[i]$ and $S[i]$ are mapped onto the processor labeled $(b_0 ... b_d)$. That is, the d most significant bits of the index of any element of the sequence are the binary representation of the label of the processor that the element resides on. This property of the mapping plays a significant role in determining the amount of communication performed during the parallel execution of the FFT algorithm. To estimate the execution time of the algorithm, each communication operation exchanges n/p words of data. The time spent in communication in the entire algorithm is $t_s \log p + t_w (n/p) \log p$. A processor updates n/p elements of R during each of the $\log n$ iterations. If a complex multiplication and addition pair takes time t_c, then the parallel run time of the binary-exchange algorithm for n-point FFT on a p-processors parallel computer is

$$T_p = t_c \frac{n}{p} \log n + t_s \log p + t_w \frac{n}{p} \log p \qquad (25)$$

So in parallel Fourier spectral method, the consuming time is about twice of T_p, which includes time for parallel FFT and time for parallel inverse FFT.

3 Numerical Results

A flexible three-dimensional solver has been written by using parallelized explicit and semi-implicit high-order upwind schemes for the spatial discretization with a high-order shock fitting algorithm. Parallel Fourier collocation method are used in computing the azimuthal direction. Investigations have been done as followings:

3.1 Parallel Performance Studies on Solving Model Equation

A two-dimensional linearized model convection-diffusion equation bounded by two parallel walls is

$$\frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} + \frac{\partial u}{\partial y} = \frac{1}{R} \frac{\partial^2 u}{\partial y^2} \qquad (26)$$

where R is the so-called "Reynolds number". The boundary conditions are $u(x,0) = u(x,1) = 0$. When R is large, there is a thin viscous boundary layer on the wall with large gradients in y direction. This model problem is not a practical flow problem, but it is used to test the accuracy and efficiency of the parallel semi-implicit method. We are looking for the temporal development of the solution in the following form:

$$u(x,y,t) = Y(y) e^{ikx} e^{-i\omega t} \qquad (27)$$

where k is a real number. The complex parameter ω and $Y(y)$ are an eigenvalue and eigenfunction of the characteristic equation. Substituting Eq. (27) into Eq. (26) leads to the following solution:

$$u_n(x,y,t) = C e^{\frac{Ry}{2}} \sin n\pi y e^{ik(x-t)} e^{-\alpha_n t} \qquad (28)$$

where $n = 1, 2, \cdots$. The solution represents and exponential decay of the oscillation energy. There, if we use $u_n(x,y,0)$ given by Eq. (28) as an initial condition, the exact solution of the model equation is also given by the same equation.

The finite difference discretization of the spatial derivatives leads to a system of semi-discrete ordinary different equations, i.e.

$$\frac{du_{ij}}{dt} = \mathbf{f}(u_{ij}) + \mathbf{g}(u_{ij}) \qquad (29)$$

where

$$\mathrm{f}(u_{ij}) = -\frac{\partial u}{\partial x} \quad (30)$$

$$\mathrm{g}(u_{ij}) = \left\{-\frac{\partial u}{\partial y} + \frac{1}{R}\left(\frac{\partial^2 u}{\partial y^2}\right)\right\}_{ij} \quad (31)$$

where explicit third-order upwind approximation is used for u_x, and second-order central difference approximation is used for u_y and u_{yy} terms. ASIRK-1 and two-sweep Gauss-Seidel line relaxation method have been used to do the temporal discretization respectively. The divide and conquer method and two kinds of parallel iterative methods (parallel Jacobi and parallel preconditioned conjugate gradient method) are used to solve a big banded matrix created by implicit methods.

A periodic boundary condition is used in the x direction. An antisymmetric boundary condition is used at the walls to calculate u located at one grid-point outside of the walls. Simple uniform grids are used. The conditions for the calculation are: $R = 10$, $k = 0.01$, $C = 1$, $n = 3$. To test the accuracy of the parallel code, the computation uses a set of 51×45 grids to discretize the computation domain. Three processors are used during computation. Figure 2 shows the contours of solutions at $t = 0.5335$. Figure 3 and Figure 4 show the numerical solution distribution along the x direction at $y = 0.18$ and the distribution along the y direction at $x = 0$ and at $t = 0.5335$. DNS results agree with exact solution very well.

To study the performance of different implicit algorithms, the execution time of an algorithm for a problem size denoted by n, on a parallel system with P processors can be written as:

$$T(n, P) = T_{calculation} + T_{communication} \quad (32)$$

Ideally, we can assume that no other overhead occurs except communication of the overlap regions, the sequential execution time is:

$$T(n, 1) = P \times T_{calculation} \quad (33)$$

Hence the speedup and parallel efficiency are given by:

$$S(n, P) = P \times \frac{T_{calculation}}{T_{calculation} + T_{communication}} = \frac{P}{1 + T_{communication}/T_{calculation}} \quad (34)$$

$$E(n, P) = \frac{1}{1 + T_{communication}/T_{calculation}} \quad (35)$$

For the best algorithm, $S(n, P) = P$, $E(n, P) = 1$, which means the ratio of $\frac{T_{communication}}{T_{calculation}}$ is close to zero. The amount of data sent and received per processor is proportional to the number of boundary cells, while the amount of computations performed by each processor is proportional to the number of interior cells. For the model problem, we have:

$$T_{calculation} = c_1 \times nx \times ny \times t_{calc} \quad (36)$$

$$T_{communication} = c_2 \times 2 \times nx \times t_{comm} \quad (37)$$

where t_{calc} represents the time required to perform a floating point operation, t_{comm} denotes the time needed to communicate one floating point number, c_1 and c_2 are constant. Figure 5 shows the efficiency comparison of different implicit algorithms by using divide and conquer method and two parallel iterative methods (Jacobi and preconditioned conjugate gradient). During the comparison, the computation size of each subdomain is keeping same along with the increasing of the computation nodes. Both divide and conquer method and parallel iterative methods perform good in both semi-implicit and full-implicit method.

3.2 Supersonic Flow over a Sphere

As the second validation of the new parallel semi-implicit code, a sphere is chosen since there are a lot of experimental results for a sphere in supersonic flow [26,27]. Our numerical results are compared with experimental results to check the accuracy of the new implicit parallel algorithms.

The body surface is a sphere given by:

$$x^{*2} + y^{*2} + z^{*2} = d^{*2} \quad (38)$$

where d^* is the radius of the sphere and is used as the reference length. The flow conditions are

$M_\infty = 5.25$ and 7.4	$\epsilon = 5 \times 10^{-4}$
$T^*_\infty = 192.989\,K$	$p^*_\infty = 10.3\,Pa$
$T^*_w = 1000\,K$	$\gamma = 1.4$
$R^* = 286.94\,Nm/kgK$	$Pr = 0.72$
$d^* = 0.6\,m$	
$T^*_r = 288\,K$	$T^*_s = 110.33\,K$
$\mu^* = 0.17894 \times 10^{-4} kg/ms$	
$Re_\infty = \rho^*_\infty U^*_\infty d^*/\mu^*_\infty = 36,159.3$	

The body surface is assumed to be a non-slip wall with an isothermal wall temperature T^*_w.

The results presented in here are obtained using 91 grid points in the streamwise direction and 61 points in the wall-normal direction. 32 Fourier collocation points are used in computing the azimuthal direction. 6 processors are using to run the high-order parallel implicit codes. Figure 6 shows the steady solution for a set of $90 \times 60 \times 32$ computational grids and the axisymmetric steady solutions for velocity vectors obtained by the numerical simulations for Mach numer 5.25 flow over a sphere. Bow shocks are captured well by the methods. Since these steady flows are axisymmetric, we present the results only in a computational surface of a fixed azimuthal angle. Figure 7 shows the contours of steady axisymmetric flow solutions of Mach numbers, pressure, and entropy for the Mach number 5.25 flow over the sphere.

Figure 8 shows the computed and experimental pressure coefficients

$$c_p = \left(p_{\text{body}} - p_\infty\right) / (\gamma/2) p_\infty {M_\infty}^2 \qquad (39)$$

for Mach number 5.25 and Mach number 7.4 in Cleary's experiments [27]. The agreement is pretty good, at least to within the scatter in the experimental data. Because of the way that the experiments were done, Cleary claimed that the data should only be accurate to $x' = 0.75$. We still can see good agreement for the larger values of x'. Figure 9 compares the computed and experimental bow shock shapes for two test cases with freestream Mach numbers of 2 and 4. Numerical results by using the new semi-implicit method compare very well with the experimental results. The new parallel semi-implicit method can approach high-order accuracy in DNS of 3-D boundary layer problem.

Grid refinement comparisons are done to verify that the results is grid independent. The numerical resolution of the $90 \times 60 \times 32$ grids is evaluated by comparing their resolutions with fine $180 \times 120 \times 32$ grid solutions for Mach number 5.25 case. Figure 10 compares the pressure coefficients on the body surface for the two sets of grids. Figure 11 shows the Mach number along the stagnation line for the two set of grids. All these steady solutions show that the two sets of results agree with each other very well and the steady solutions are well resolved by the grids.

3.3 Receptivity of 3-D Hypersonic Boundary Layer over a Blunt Wedge

Our new 3-D parallel shock-fitting hypersonic DNS code has been applied to DNS of the receptivity of a three-dimensional boundary layer to weak freestream oblique acoustic disturbance waves for a hypersonic flow boundary layer over a 3-D parabolic leading edge whose body surface is given by

$$x^* = b^* {y^*}^2 - d^* \qquad (40)$$

where b^* is a given constant and d^* taken as the reference length. The body surface is assumed to be a non-slip wall with an isothermal wall temperature T_w^*.

The specific flow conditions are

$M_\infty = 15$	$\epsilon = 5 \times 10^{-4}$
$T_\infty^* = 192.989\,K$	$p_\infty^* = 10.3\,Pa$
$T_w^* = 1000\,K$	$\gamma = 1.4$
$R^* = 286.94\,Nm/kgK$	$Pr = 0.72$
$b^* = 40\,m^{-1}$	$d^* = 0.1\,m$
$T_r^* = 288\,K$	$T_s^* = 110.33\,K$
$\mu^* = 0.17894 \times 10^{-4} kg/ms$	
Nose Radius of Curvature $= r^* = 0.0125\,m$	
$Re_\infty = \rho_\infty^* U_\infty^* d^* / \mu_\infty^* = 6026.55$	

There is no flow in spanwise direction. Steady results have been obtained by using the new implicit parallel shock-fitting code.

In the simulations, steady flow solutions are first obtained by advancing the unsteady flow computations to convergence using the new parallelized semi-implicit computer code. No disturbances are imposed in the freestream. Subsequently, freestream disturbances are superimposed on the steady mean flow to investigate the development of T-S waves in the boundary layer with the effects of the bow shock interaction. The freestream disturbances are assumed to be weak monochromatic planar acoustic waves with wave front oblique to the center line of the body in the $x - z$ plane at an angle of ψ. The perturbations of flow variables in the freestream introduced by the freestream acoustic wave before reaching the bow shock can be written in the following form:

$$q'_\infty \;=\; |q'|_\infty \, e^{i(k\cos\psi\, x + k\sin\psi\, z - \omega t)} \qquad (41)$$

where $|q'|$ represents one of the flow variables, $|u'|$, $|v'|$, $|w'|$, $|p'|$, and $|\rho'|$. The freestream perturbation amplitudes satisfy the following relations:

$$|u'|_\infty = \epsilon\cos\psi\,, \qquad |v'|_\infty = 0\,,$$
$$|p'|_\infty = \gamma M_\infty \epsilon\,, \qquad |\rho'|_\infty = M_\infty \epsilon$$
$$|w'|_\infty = \epsilon\sin\psi$$

where ϵ represents the freestream wave magnitude. The angle ψ is the angle of freestream wave with respect to the x axis in the x-z plane, where $\psi = 0^o$ corresponds to 2-D planar waves. The parameter k is the dimensionless freestream wave number which is related to the dimensionless circular frequency ω by:

$$\omega = k\,(\cos\psi + M_\infty^{-1}) \qquad (42)$$

The dimensionless frequency F is defined as:

$$F = \frac{\omega^* \nu^*}{U_\infty^{*2}}\,. \qquad (43)$$

Steady Flow Solutions

The specific flow conditions are

$M_\infty = 15$	$\epsilon = 5 \times 10^{-4}$
$T_\infty^* = 192.989\,K$	$p_\infty^* = 10.3\,Pa$
$T_w^* = 1000\,K$	$\gamma = 1.4$
$R^* = 286.94\,Nm/kgK$	$Pr = 0.72$
$b^* = 40\,m^{-1}$	$d^* = 0.1\,m$
$T_r^* = 288\,K$	$T_s^* = 110.33\,K$
$\mu^* = 0.17894 \times 10^{-4} kg/ms$	
Nose Radius of Curvature $= r^* = 0.0125\,m$	
$Re_\infty = \rho_\infty^* U_\infty^* d^* / \mu_\infty^* = 6026.55$	

The body surface is assumed to be a non-slip wall with an isothermal wall temperature T_w^*.

To show the efficiency and accuracy of the code using high-order semi-implicit method, we compare the results of the steady flow solutions of the Navier-Stokes equations for the viscous hypersonic flow over the 3-D

blunt wedge obtained by using this new method with the results obtained by using the parallelized explicit fifth-order unsteady computer code in Ref. [21]. We use ASIRK-1C method in time steps and advance the solutions to a steady state without freestream perturbations.

The results presented in here are obtained by using 161 grid points in the streamwise direction and 121 points in the wall-normal direction, 4 Fourier collocation points are used in computing the azimuthal direction. Figure 12 shows the steady solution for a set of $161 \times 121 \times 4$ computational grids obtained by the simulation. Figure 13 shows steady flow solutions for temperature and pressure contours obtained by using 12 processors simultaneously. Edges in figure are the boundaries of subdomains.

Unsteady Flow Solutions

In this section, we choose the generation of boundary-layer T-S and inviscid instability waves by freestream acoustic disturbances for hypersonic flow over a parabolic leading edge with freestream disturbance frequency $F = 1770$, and $\epsilon = 5 \times 10^{-3}$ as the test case for comparing the new parallelized semi-implicit method using divide and conquer method solving banded Jacobian matrix with the parallelized explicit method. We change the freestream disturbance wave angle ψ from 0^0 to 15^0, 30^0, 45^0, 55^0, 60^0.

Figure 15 shows the 3-D contours of the instantaneous perturbation u' the velocity in x direction for $\psi = 30^0$. Figure 14 shows contours of the instantaneous perturbation u' the velocity in one cross-section and the Fourier amplitude $\mid u' \mid$ after the flow field has reached a periodic state for $\psi = 30^0$. The instantaneous contours of u' show the development of three-dimensional first-mode waves in the boundary layer on the surface. From the figure, the characteristics of the switching of instability modes for first mode instability dominated region to second mode with the sudden phase angle change near the body surface around $x = 0.2$ can be observed. Table 1 shows the efficiency comparisons of the CPU time consuming between parallel explicit method and parallel semi-implicit method on the simulation of hypersonic boundary layer over 3-D wedge computations for different cases using different grids. The efficiency can be improved by using the new implicit parallel method.

Figure 16 shows the comparison of distribution of the Fourier amplitude $\mid u' \mid$ along x axis. Figure 17 shows the comparison of the maximum Fourier amplitude $\mid u' \mid$ of the first mode. For the first mode, the maximum Fourier amplitude occurs around $\psi = 45°$. But for the second mode, the maximum Fourier amplitude occurs on the 2-D wave case of $\psi = 0°$. This is same as the conclusion of Mach's work [28], where the 3-D effects on linear stability of compressible flat plate boundary layer were studied.

Table 1: **Comparison of full explicit method with semi-implicit method for hypersonic boundary layer over 3-D wedge computations.(Zone 1 case)**

	Explicit Method	Semi-implicit	Ratio
CFL number	0.012	0.93	–
$\frac{\Delta x}{\Delta y}$	102.04	102.04	–
timesteps	201,640	3,000	–
time(sec)	0.5951×10^{-2}	0.5951×10^{-2}	–
CPU time(sec)	14,728.77	1,644.57	8.956

3.4 Simulations of Nonlinear Wave Breakdown of 3-D Hypersonic Flows (On-going)

The new efficient parallel implicit codes for massively parallelized computers are used to do the further direct numerical simulations of nonlinear wave breakdown simulation of 3-D hypersonic flows. 2-D hypersonic mean flow over a parabola has been obtained by using new parallel implicit code. The surface of the parabola has been described in Eq. (40). The body surface is assumed to be a non-slip wall with an isothermal wall temperature T_w^*.

The specific flow conditions are

$M_\infty = 15$ $\quad \epsilon = 5 \times 10^{-4}$
$T_\infty^* = 192.989\,K$ $\quad p_\infty^* = 10.3\,Pa$
$T_w^* = 1000\,K$ $\quad \gamma = 1.4$
$R^* = 286.94\,Nm/kgK$ $\quad Pr = 0.72$
$b^* = 40\,m^{-1}$ $\quad d^* = 0.1\,m$
$T_r^* = 288\,K$ $\quad T_s^* = 110.33\,K$
$\mu^* = 0.17894 \times 10^{-4} kg/ms$
Nose Radius of Curvature $= r^* = 0.0125\,m$

Figure 18 and Figure 19 show the Mach number contours and temperature contours of 2-D mean flow for this test case. 1600×241 grids are used in the whole computation domain.

Before we start to do the numerical simulation of non-linear wave breakdown of 3-D hypersonic flow, necessary linear stability (LST) analysis is carried out to obtain more clear knowledgement of the characteristics of the wave used in the simulation. The linear stability analysis shares the same basic flow solutions with the DNS simulation. Since the linear stability analysis has to be conducted station by station, the nondimensionalization is done with respect to local shock layer edge values. The Reynolds number used in the calculations is the local Reynolds number R based on the local length scale. A more standard length scale $\delta(s)$ is used to scale the wave numbers and frequencies. $\delta(s)$ is defined as

$$\delta = \sqrt{\frac{s\nu_e}{U_e}} \tag{44}$$

where s is the distance from the leading dege. The Reynolds number based on this length scale δ is R. A non-dimensional frequency F and the dimensionless circular frequency ω are also introduced and defination can be found in previous section.

A common forcing frequency is enforced in DNS. This frequency is also enforced locally at each station in the linear stability analysis during the study. In the normal mode analysis for the linear disturbances, the fluctuations of flow quantities are assumed to be represented by harmonic waves of the following form:

$$[u', v', p', T', w']^T = \left[\hat{u}(y), \hat{v}(y), \hat{p}(y), \hat{T}(y), \hat{w}(y)\right]^T e^{i(\alpha x + \beta z - \omega t)} \quad (45)$$

where α and β are the wavenumbers in x and z directions respectively, and ω is the frequency of the disturbance waves. These parameters are in general complex numbers. The complex amplitude (eigen) function of a typical flow variable, say u, is $\hat{u}(y)$. In order to compare with the DNS results, spatially stability problem is solved. In a spatial stability problem, real-valued ω and β are assumed. While α is the complex eigenvalue to be solved for. The real part of α, α_r, represents the spatial frequency of the disturbance modes, while the imaginary part, α_i, represents the spatial amplification rate of the disturbances. When $-\alpha_i$ is greater, equal to, or smaller than zero, a disturbance mode is unstable with finite amplification, neutrally stable, or stable with finite damping, respectively. In our research, a range of forcing frequency is studied for obtaining spatial amplification rate of the disturbances.

To get secondary instability on higher Reynolds number region, lower frequence $F = 223$ are forced in LST. Figure 20 shows the amplification rate path of first mode and second mode from LST with frequence $F = 223$. The reference length scale is the local length scale δ. It is clear that LST predicts the change of dominance of the boundary layer modes in the streamwise direction from the first modes to higher modes. There is a short range where the second modes become unstable at higher Reynolds number. The highest $-\alpha_i$ occurs at about $R = 442$. Then we can fix the station of $R = 417$ and change F to get different $-\alpha_i$. The results are shown in Figure 21. From this figure, the highest $-\alpha_i$ occurs when F is set to 198.

These studies gives us very good understandings of the characteristics of the wave. Then we can either introduce disturbances into the boundary layer by blowing and suction within a narrow disturbance strip in the leading edge region of the wall or impose the eigen function of the unstable second mode into the inlet of the flow region to carry out the direct numerical simulations of nonlinear wave breakdown of hypersonic boundary layer over 3-D wedges by setting the frequency of wave to the frequency of the most unstable wave in our study. Further unsteady calculation will be continued soon.

Summary

We have developed a flexible high-order parallelized computer codes which include parallel Fourier spectral method for the direct simulations of fully 3-D hypersonic boundary layers over a blunt body to freestream acoustic disturbance using Navier-Stokes equations. Divide and conquer methods are chosen to solve the banded matrix parallelizedly because of its good performance. Parallel iterative methods are considered too. From the results of studying convection-diffusion model equation, the supersonic flow over a sphere and hypersonic boundary layer receptivity over a blunt wedge, the new algorithms and computer codes reduced the computational CPU times by an order of magnitude while maintaining high accuaracy. The results shows that it is feasible for us to use new efficient parallel implicit code to do further study on nonlinear wave breakdown of hypersonic boundary layer over 3-D blunt bodies on current computation conditions.

Acknowledgments

This research was supported by the Air Force Office of Scientific Research under grant numbers F49620-00-1-0101 monitored by Dr. Robert Herklotz.

References

[1] Board, D. S., "Final Report of the Second Defense Science Board Task Force on the National Aero-Space Plane (NASP)," *AD-A274530, 94-00052, November*, 1992.

[2] Kleiser, L. and Zang, T. A., "Numerical Simulation of Transition in Wall-Bounded Shear Flows," *Ann. Rev. Fluid Mech.*, Vol. 23, 1991, pp. 495–535.

[3] Reed, H. L., "Direct Numerical Simulation of Transition: the Spatial Approach," *Progress in Transition Modeling*, AGARD-Report-793 1994.

[4] Guo, Y., Adams, N. A., Sandham, N. D., and Kleiser, L., "Numerical Simulation of Supersonic Boundary Layer Transition," *Application of Direct and Large Eddy Simulation to Transition and Turbulence*, AGARD-CP-551, 1994.

[5] Pruett, C. D., Zang, T. A., Chang, C.-L., and Carpenter, M. H., "Spatial Direct Numerical Simulation of High-Speed Boundary-Layer Flows, Part I: Algorithmic Considerations and Validation," *Theoretical and Computational Fluid Dynamics*, Vol. 7, 1995, pp. 49–76.

[6] Lele, S. K., "Compact Finite Difference Schemes with Spectral-like Resolution," *Journal of Computational Physics*, Vol. 103, 1992, pp. 16–42.

[7] Abdallah, S., "Parallel Processing for Implicit Solutions of the Compressible Navier-Stokes Equations," *AIAA Journal*, Vol. 32, No. 12, pp. 2469–2471, 1994.

[8] Jameson, A. and Yoon, S., "Lower-Upper Implicit Schemes with Multiple Grids for the Euler Equations," *AIAA Journal*, Vol. 25, No. 7, pp. 929–935, 1987.

[9] Tidriri, M. D., "Domain Decomposition for Compressible Navier-Stokes Equations with Different Discretizations and Formulations," *Journal of Computational Physics*, Vol. 119, 1995, pp. 271-282.

[10] A. Averbuch, L. Ioffe, M. I. L. V., "Two-dimensional parallel solver for the solution of Navier-Stokes Equations with Constant and Variable Coefficients Using ADI on Cells," *Parallel Computing*, Vol. 24, pp.673–699, 1998.

[11] A. Basermann, B. R. and Schelthoff, C., "Preconditioned CG Methods for Sparse Matrices on Massively Parallel Machines," *Parallel Computing*, Vol. 23, pp. 381–398, 1997.

[12] M. A. Lambert, G. H. R. and Hewett, D. W., "A Parallel DSDADI Method for Solution of the Steady State Diffusion Equation," *Parallel Computing*, Vol. 23, pp. 2041–2065, 1997.

[13] Brugnano, L., "A Parallel Solver for Tridiagonal Linear Systems for Distributed Memeory Parallel Computers," *Parallel Computing*, Vol. 17, pp. 1017–1023, 1991.

[14] Brass, A. and Pawley, G. S., "Two and three dimensional FFTs on highly parallel computers," *Parallel Computing*, Vol. 3, pp. 167–184, 1986.

[15] Pelz, R., "The parallel Fourier pseudospectral method," *Journal of Computational Physics*, Vol. 92, pp. 296–312, 1991.

[16] Calvin, C., "Implementation of parallel FFT algorithms on distributed memory machines with a minimum overhead of communication," *Parallel Computing*, Vol. 22, pp. 1255–1279, 1996.

[17] A. Averbuch, M. I. and Vozovoi, L., "Parallel Implementation of Non-linear Evolution Problems Using Parabolic Domain Decomposition," *Parallel Computing*, Vol. 21, pp. 1151–1183, 1995.

[18] Zhong, X., "Direct Numerical Simulation of Hypersonic Boundary-Layer Transition Over Blunt Leading Edges, Part I: New Numerical Methods and Validation ," AIAA paper 97-0755, Jan. 1997.

[19] Zhong, X., "Direct Numerical Simulation of 3-D Hypersonic Boundary Layer Receptivity to Freestream Disturbances," *AIAA paper 98-0533, 36th AIAA Aerospace Sciences Meeting and Exhibit, January 12-15, Reno, Nevada*, 1998.

[20] Zhong, X., "High-Order Finite-Difference Schemes for Numerical Simulation of Hypersonic Boundary Layer Transiton," *Journal of Computational Physics*, Vol. 144, August 1998, pp. 662-709.

[21] Zhong, X. and Dong, H., "Hypersonic Boundary-Layer Receptivity to Freestream Disturbances over an Elliptical Cross-Section Cone," *AIAA paper 99-0409*, January 1999.

[22] Dong, H. and Zhong, X., "A Parallel High-Order Implicit Algorithm for Compressible Navier-Stokes Equations," *AIAA paper 2000-0275*, January 2000.

[23] Zhong, X., "Additive Semi-Implicit Runge-Kutta Schemes for Computing High-Speed Nonequilibrium Reactive Flows," *Journal of Computational Physics*, Vol. 128, 1996, pp. 19-31.

[24] Yoh, J. J. and Zhong, X., "Low-Storage Semi-Implicit Runge-Kutta Methods for Reactive Flow Computations," AIAA paper 98-0130, Jan. 1998.

[25] Dong, H. and Zhong, X., "High-Order Semi-Implicit Simulation of Hypersonic Boundary Layer Stability and Transition," *AIAA paper 98-0127, 36th AIAA Aerospace Sciences Meeting and Exhibit, January 12-15, Reno, Nevada*, 1998.

[26] Kopriva, D. A., "Multidomain Spectral Solutions of High-Speed Flows over Blunt Cones," *AIAA Journal*, Vol. 31, No. 12, 1993.

[27] Cleary, J. W., "An Experimental and Theoretical Investigation of the Pressure Distribution and Flowfields of Blunted Cones at Hypersonic Mach Numbers," NASA TN D-2969 1965.

[28] Mack, L. M., "Boundary Layer Linear Stability Theory," AGARD report, No. 709, 1984.

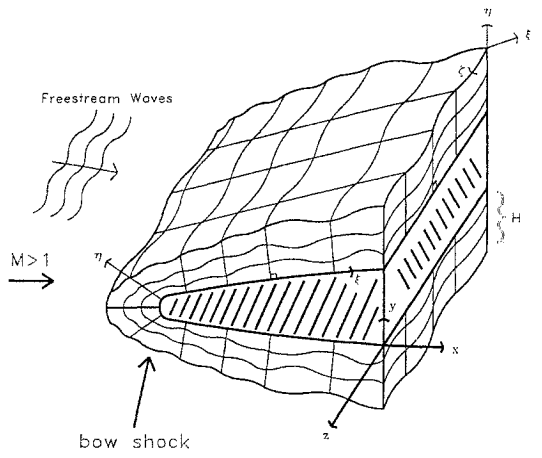


Figure 1: A schematic of 3-D shock fitted grids for the direct numerical simulation of hypersonic boundary-layer receptivity to freestream disturbances over a blunt leading edge.

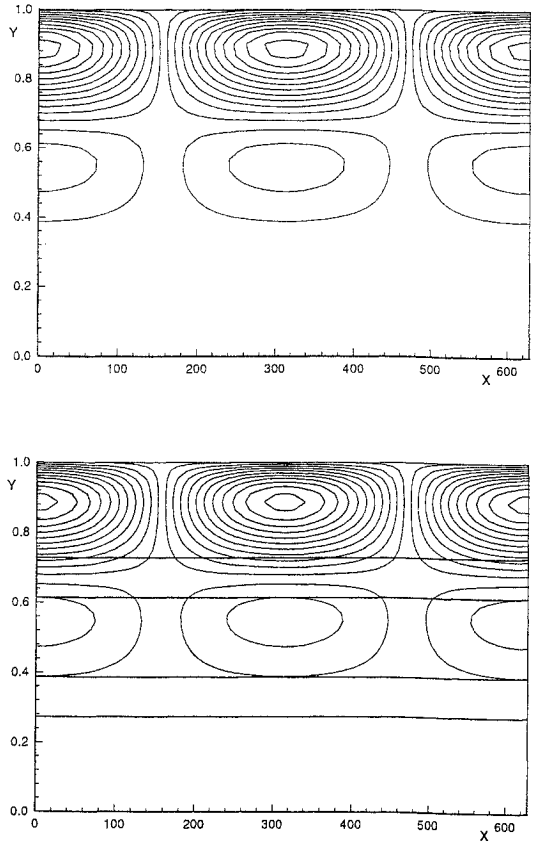


Figure 2: Contours of instantaneous solution at $t = 0.5335$, exaction solution (upper figure), and parallelized semi-implicit solution (lower figure) by using 3 processors.

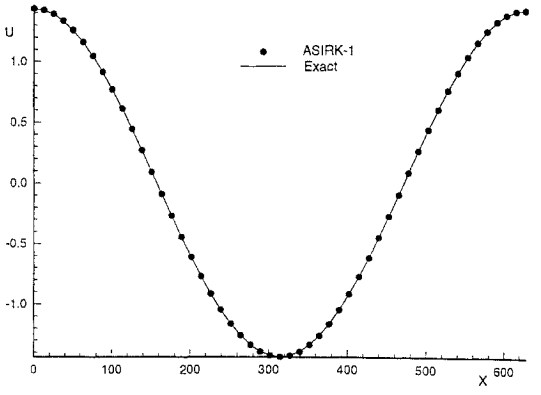


Figure 3: Comparisons of the distribution of transient solution in x direction between numerical result and exact solution ($y = 0.16$, $t = 0.5335$).

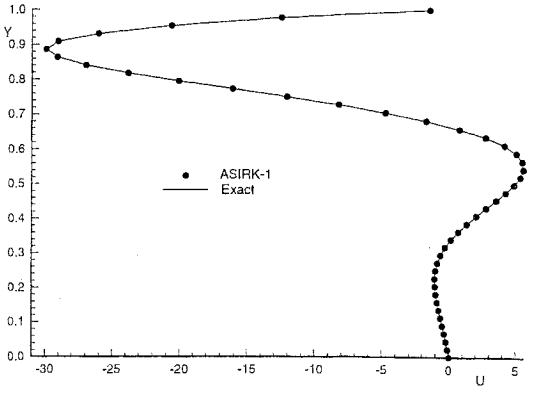


Figure 4: Comparisons of the distribution of transient solution in y direction between numerical result and exact solution ($x = 0$, $t = 0.5335$).

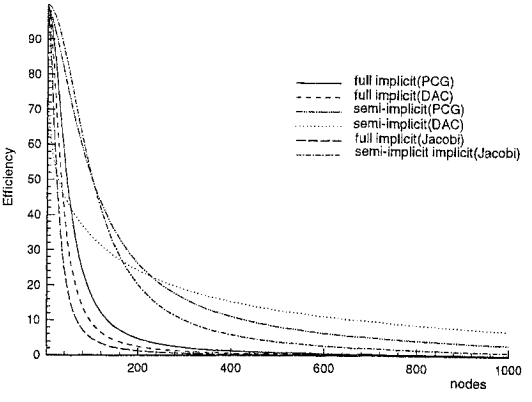


Figure 5: Efficiency comparisons of implicit algorithms by using divide and conquer(DAC) method and parallel iterative methods (Jacobi and preconditioned conjugate gradient(PCG)).

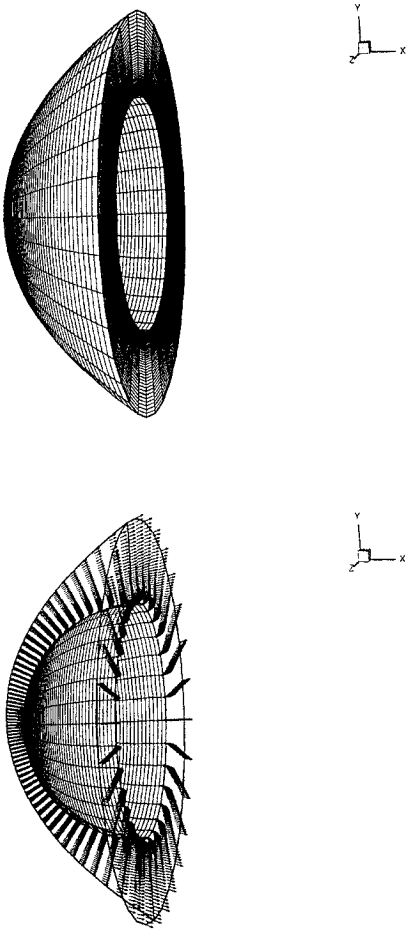


Figure 6: Steady flow solutions for $M_\infty = 5.25$ flow over a sphere by using high order 3-D shock-fitting method for computational grids (upper figure) where the bow shock shape is obtained as the freestream grid line, and velocity vectors (lower figure).

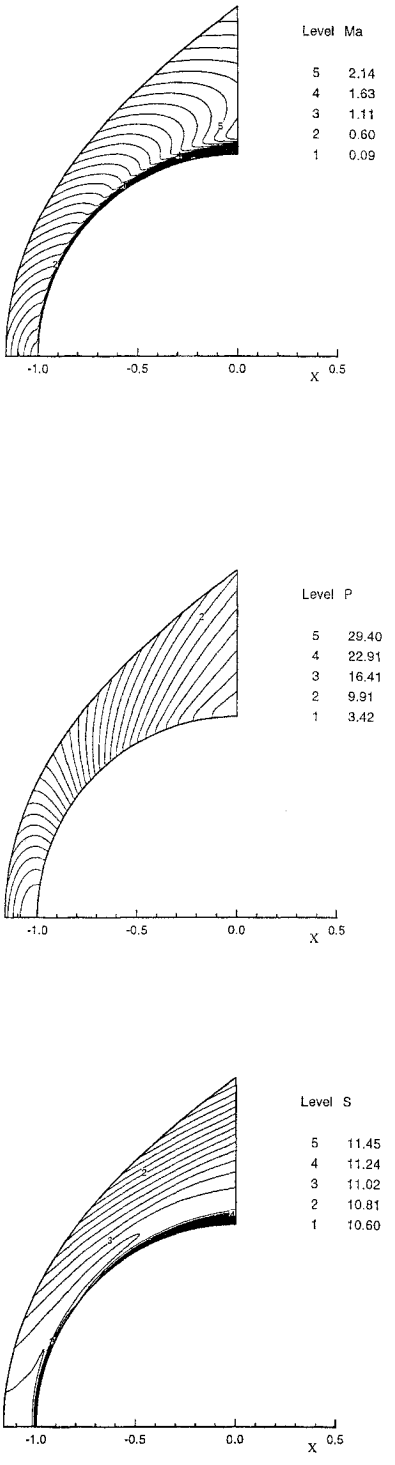
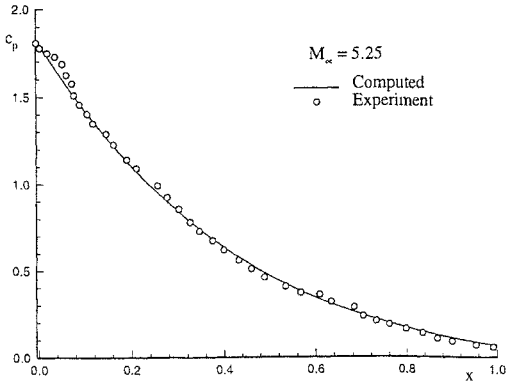


Figure 7: Contours of axisymmetric steady base flow solutions over a sphere behind the bow shock with $M_\infty = 5.25$: Mach numbers (upper figure), pressure (middle figure), and entropy (lower figure).



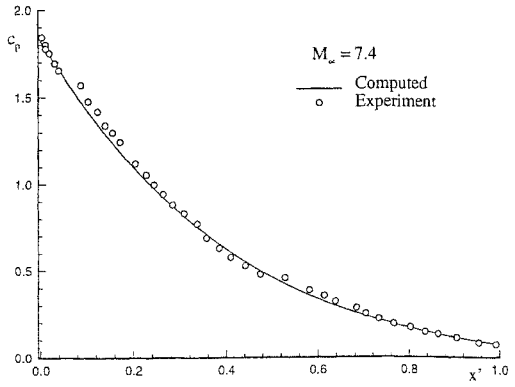


Figure 8: Comparison of pressure coefficients on a spherical body with different Mach numbers.

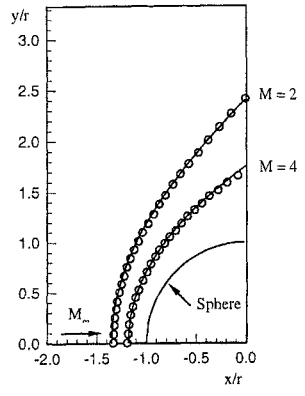


Figure 9: Bow shock locations for hypersonic flow over a sphere (lines: Numerical solutions, circles: experimental results).

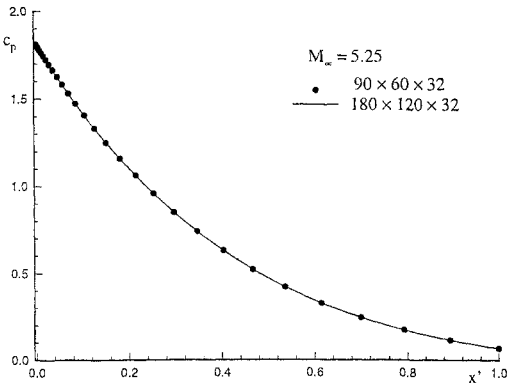


Figure 10: Comparison of steady solution of the pressure coefficients along the sphere surface for two sets of grids.

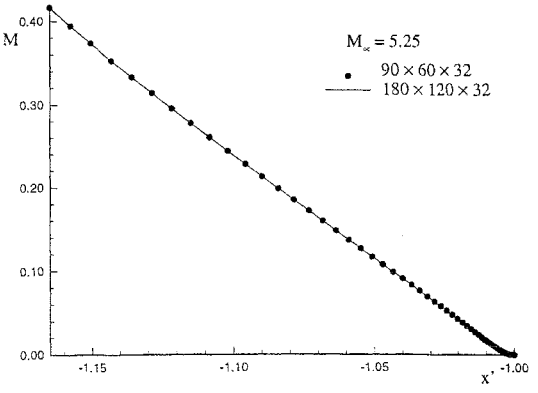


Figure 11: Comparison of steady solution of the Mach number along the stagnation line for two sets of grids.

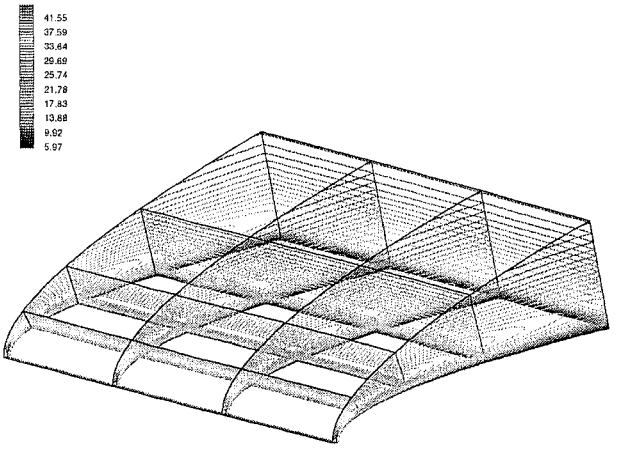


Figure 12: Steady flow solution of temperature contours for hypersonic flow over a blunt wedge by using high order parallelized code.

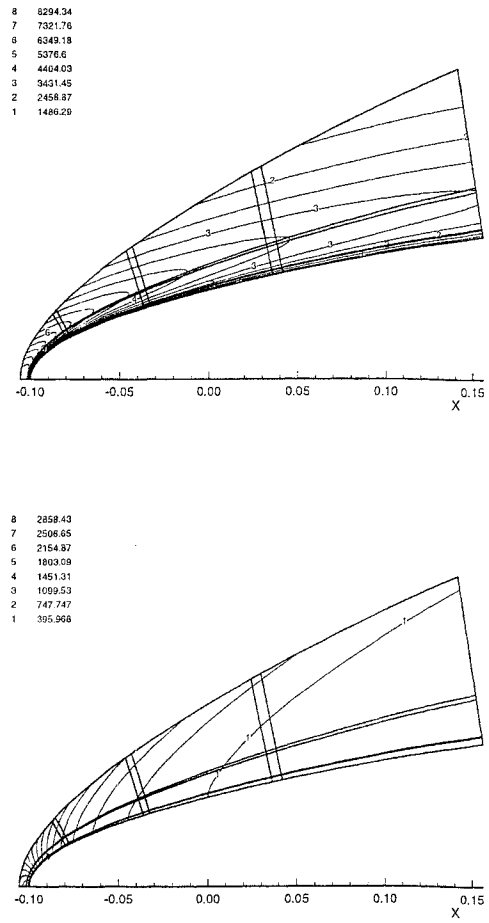


Figure 13: Steady flow solutions for hypersonic flow over a blunt wedge by using high order parallelized code for temperature (upper figure) and pressure contours (lower figure). Edges show the boundaries of each subdomain.

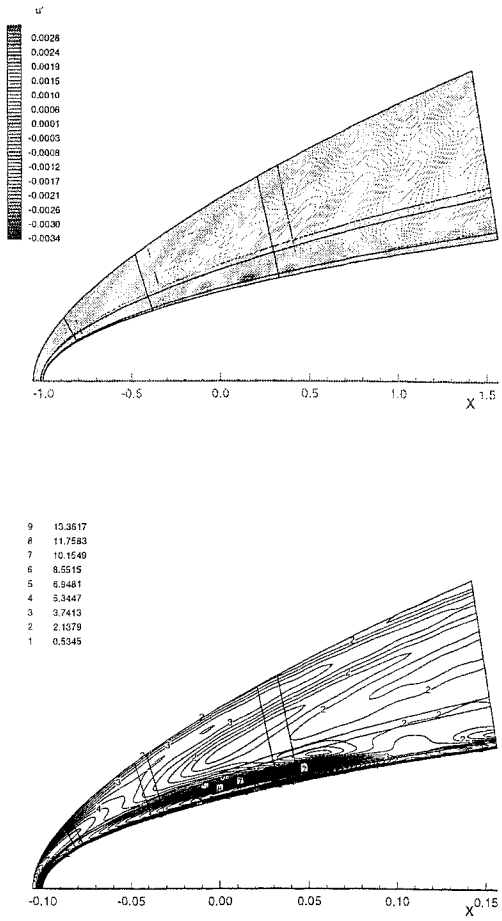


Figure 14: Instantaneous u' contours for the receptivity to freestream disturbances for 3-D hypersonic boundary-layer over a parabolic leading edge. $\psi = 30^0$ (upper figure) and Fourier amplitude $\mid u' \mid$ (lower figure) Edges show the boundaries of each subdomain.

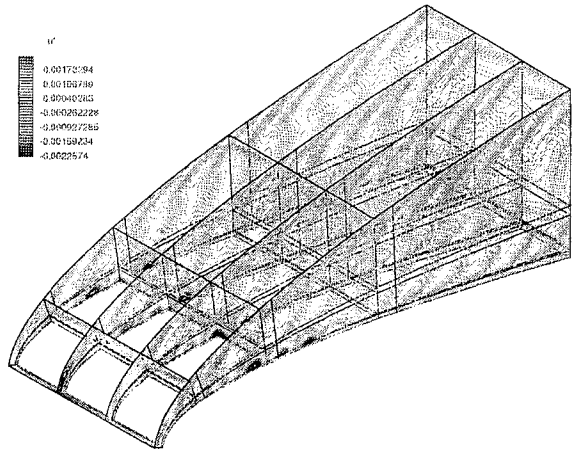


Figure 15: Instantaneous u' contours for the receptivity to freestream disturbances for 3-D hypersonic boundary-layer over a parabolic leading edge ($\psi = 30^0$). Edges show the boundaries of each subdomain.

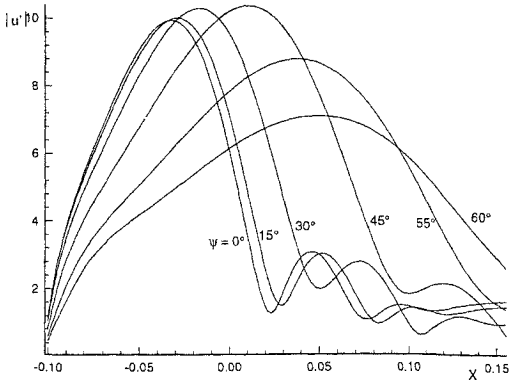


Figure 16: Comparison of the distribution of Fourier amplitude $\mid u' \mid$ along x axis for different ψ cases.

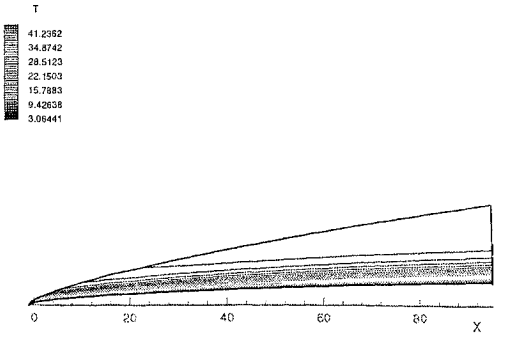


Figure 19: Temperature contours of Mach number 15 hypersonic flow over longer wedge.

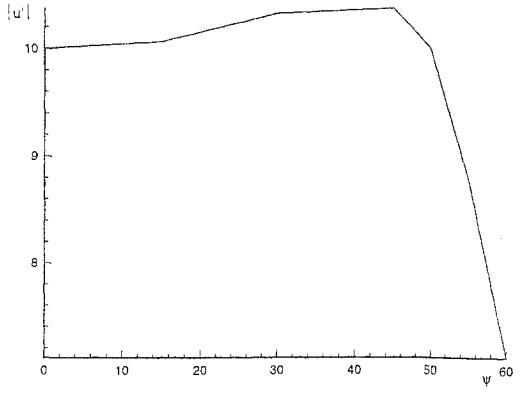


Figure 17: Comparison of maximum Fourier amplitude $\mid u' \mid$ for different ψ cases.

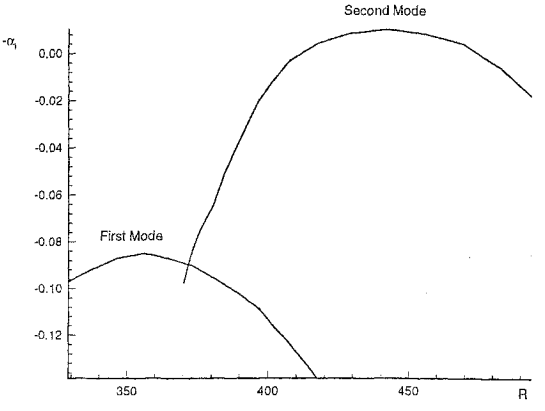


Figure 20: Amplification rate path of LST modes at $F = 223$.

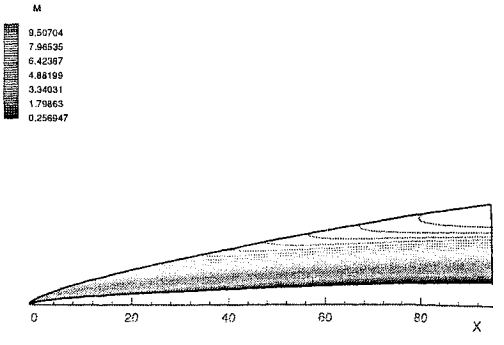


Figure 18: Mach number contours of Mach number 15 hypersonic flow over longer wedge.

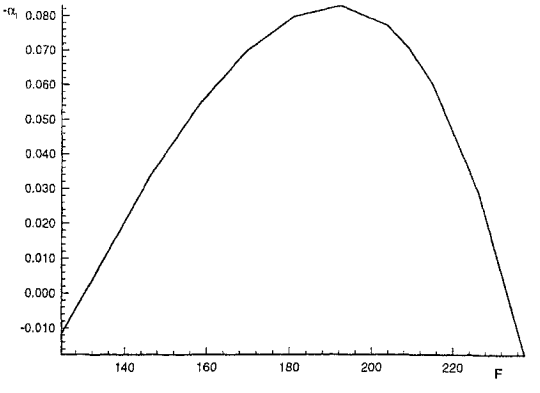


Figure 21: Amplification rate of second mode along with frequence changing at $R = 442$.