JOURNAL OF SPACECRAFT AND ROCKETS
Vol. 40, No. 2, March–April 2003

New Forward/Backward Sweeping Parabolized Navier–Stokes Algorithm

Hiromasa Kato* and John C. Tannehill†
Iowa State University, Ames, Iowa 50011

A new forward/backward sweeping parabolized Navier–Stokes algorithm has been developed to compute efficiently supersonic/hypersonic flowfields with embedded separated regions. The algorithm splits the streamwise flux vector using the Steger–Warming method and employs multiple forward/backward sweeps of the flowfield to duplicate the results that would be obtained with the complete Navier–Stokes equations. The forward/backward sweeping of the flowfield significantly reduces the number of iterations required over previous iterative parabolized Navier–Stokes algorithms. Once a separated flow region is computed, the algorithm returns to the usual forward-space-marching mode until the next separated flow region is encountered. The algorithm has been successfully incorporated into NASA's parabolized Navier–Stokes UPS code. The new algorithm has been applied to three separated flow test cases consisting of flow over a compression ramp and two flows over a hollow-cylinder-flare geometry. The present numerical results are in excellent agreement with complete Navier–Stokes computations and experimental data.

Nomenclature

A	=	flux Jacobian matrix
C_f	=	skin-friction coefficient
C_p	=	pressure coefficient
$d\boldsymbol{S}$	=	cell face area vector
$\boldsymbol{E}, \boldsymbol{F}, \boldsymbol{G}$	=	flux vectors
E_t	=	total energy
e	=	specific internal energy
J	=	Jacobian of transformation
L	=	reference length
M	=	Mach number
M_ξ	=	local streamwise Mach number
Pr	=	Prandtl number
p	=	static pressure
q_x, q_y, q_z	=	heat flux components
R	=	gas constant
Re	=	Reynolds number
St	=	Stanton number
T	=	static temperature
$[T]$	=	right eigenvector matrix
t	=	time
$\boldsymbol{U}$	=	vector of conserved variables
u, v, w	=	Cartesian velocity components
V	=	speed of flow
β	=	safety factor
γ	=	ratio of specific heats
Δt	=	time increment
$\Delta \boldsymbol{U}_i$	=	$\boldsymbol{U}_{i+1} - \boldsymbol{U}_i$
$\Delta^- \boldsymbol{U}_i$	=	$\boldsymbol{U}_{i-1} - \boldsymbol{U}_i$
$\Delta\xi$	=	streamwise spatial increment
μ	=	viscosity coefficient
ξ, η, ζ	=	transformed space coordinates
ρ	=	density
τ	=	viscous stress
ω	=	Vigneron et al.[6] parameter

Subscripts

i	=	inviscid flux term or streamwise spatial index
v	=	viscous flux term
w	=	wall value
ξ, η, ζ	=	$\partial/\partial\xi, \partial/\partial\eta, \partial/\partial\zeta$
∞	=	freestream value

Superscripts

n	=	iteration (time) level index
p	=	Vigneron et al.[6] split flux (negative eigenvalues)
$*$	=	Vigneron et al.[6] split flux (positive eigenvalues)
$+$	=	Steger–Warming[7] split flux (positive eigenvalues)
$-$	=	Steger–Warming[7] split flux (negative eigenvalues)
$'$	=	viscous flux without streamwise derivatives

Presented as Paper 2002-0735 at the AIAA 40th Aerospace Sciences Meeting, Reno, NV, 14–17 January 2002; received 17 February 2002; revision received 7 October 2002; accepted for publication 12 November 2002.

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

†Manager, Computational Fluid Dynamics Center, and Professor, Department of Aerospace Engineering and Engineering Mechanics. Fellow AIAA.

Introduction

SUPERSONIC/HYPERSONIC separated flowfields are normally computed using a complete Navier–Stokes (NS) solver. Recently, a new numerical approach has been developed[1–5] that iteratively solves the parabolized NS (PNS) equations. In this approach, the standard single-sweep PNS method is used to march the solution in the streamwise direction in regions where there are negligible upstream influences. In regions where upstream influences are present (such as near flow separations), the governing equations are solved using multiple streamwise sweeps to duplicate the results that would be obtained with the complete NS equations. As a result of this approach, an entire flowfield can be computed much more efficiently (in terms of computer time and storage) than with a standard NS solver, which marches the entire solution in time using the time-dependent approach.

Both an iterative PNS (IPNS) and a time-iterative PNS (TIPNS) algorithm have been developed previously. The IPNS algorithm[1–3] splits the streamwise flux vector using Vigneron et al. splitting[6] and can be applied to flows with moderate upstream influences and small streamwise separated regions. The TIPNS algorithm[4,5] splits the streamwise flux vector using Steger–Warming splitting[7] and may retain the time-derivative terms. The TIPNS algorithm can be used to compute flows with strong upstream influences including large streamwise separated regions. The TIPNS approach has been used to compute successfully hypersonic separated flows over two-dimensional compression ramps[8] and axisymmetric hollow-cylinder-flare geometries.[9]

In the previous IPNS/TIPNS algorithms, forward-space marching is used exclusively to iterate the regions with significant upstream influence. Because the flowfield is predominantly supersonic in a

typical application, the majority of the acoustic information is propagated downstream. Hence, the choice of using a forward-sweep iteration is a natural one, and it also provides a very straightforward method of extending the already robust PNS framework. With this approach, however, the upstream influence can only be propagated one streamwise station per iteration. In many cases this does not pose a serious problem because the regions with upstream influence are relatively small compared to the overall size of the solution domain. On the other hand, if the relative size of the upstream influence region is large, the forward-sweep TIPNS method requires a significant number of iterations to converge. Therefore, it is desirable to design a scheme such that the upstream influence is propagated much faster while maintaining the current PNS/IPNS/TIPNS-type space-marching structure.

In the present study, the TIPNS method has been extended to permit forward/backward alternating direction marching. In this approach, the iterated region is computed using forward and backward sweeps, alternating the marching direction after every global sweep (iteration). When backward sweeping is incorporated in the process, upstream influences are propagated over the entire iterated region in a single sweep. This in turn results in a significant acceleration of the convergence of the solution. Previous attempts[10–12] at accelerating the convergence of multiple-sweep PNS methods have produced limited results and are not as efficient as the present approach. In the approach of Barnett and Davis,[10] only the pressure is propagated during the backward sweep. In Refs. 11 and 12, the solution is iterated in time at every streamwise station (during each sweep), which significantly increases the computation time.

The new forward/backward sweeping iterative PNS (FBIPNS) algorithm has been applied to three separated flow test cases. These test cases include the two-dimensional supersonic laminar flow over an 8-deg compression ramp and two hypersonic laminar flows over an axisymmetric hollow-cylinder-flare geometry.[13] The present numerical results are compared with results obtained using the OVERFLOW[14] and LAURA[15] NS codes and with experimental results obtained by Holden.[13]

Governing Equations

The thin-layer NS equations are obtained from the compressible NS equations by dropping all viscous terms except the normal viscous terms. If the crossflow viscous terms are also retained, the resulting equations can be written in a general nonorthogonal coordinate system (ξ, η, ζ) as

$$\frac{1}{J}\frac{\partial \boldsymbol{U}}{\partial t} + \frac{\partial \boldsymbol{E}}{\partial \xi} + \frac{\partial \boldsymbol{F}}{\partial \eta} + \frac{\partial \boldsymbol{G}}{\partial \zeta} = 0 \tag{1}$$

where J is the Jacobian of the transformation and

$$\boldsymbol{U} = [\rho, \rho u, \rho v, \rho w, E_t]^T$$

$$\boldsymbol{E} = (\xi_x/J)\boldsymbol{E}_i + (\xi_y/J)\boldsymbol{F}_i + (\xi_z/J)\boldsymbol{G}_i$$

$$\boldsymbol{F} = (\eta_x/J)(\boldsymbol{E}_i - \boldsymbol{E}_v') + (\eta_y/J)(\boldsymbol{F}_i - \boldsymbol{F}_v') + (\eta_z/J)(\boldsymbol{G}_i - \boldsymbol{G}_v')$$

$$\boldsymbol{G} = (\zeta_x/J)(\boldsymbol{E}_i - \boldsymbol{E}_v') + (\zeta_y/J)(\boldsymbol{F}_i - \boldsymbol{F}_v') + (\zeta_z/J)(\boldsymbol{G}_i - \boldsymbol{G}_v')$$

The prime in the preceding equations indicates that the streamwise (ξ direction) viscous terms have been dropped. These same viscous terms are also dropped in the PNS equations and the boundary-layer equations. The inviscid (subscript i) and viscous (subscript v) flux vectors are given by

$$\boldsymbol{E}_i = [\rho u, \rho u^2 + p, \rho uv, \rho uw, (E_t + p)u]^T$$

$$\boldsymbol{F}_i = \left[\rho v, \rho uv, \rho v^2 + p, \rho vw, (E_t + p)v\right]^T$$

$$\boldsymbol{G}_i = \left[\rho w, \rho uw, \rho vw, \rho w^2 + p, (E_t + p)w\right]^T$$

$$\boldsymbol{E}_v = [0, \tau_{xx}, \tau_{xy}, \tau_{xz}, u\tau_{xx} + v\tau_{xy} + w\tau_{xz} - q_x]^T$$

$$\boldsymbol{F}_v = [0, \tau_{yx}, \tau_{yy}, \tau_{yz}, u\tau_{yx} + v\tau_{yy} + w\tau_{yz} - q_y]^T$$

$$\boldsymbol{G}_v = [0, \tau_{zx}, \tau_{zy}, \tau_{zz}, u\tau_{zx} + v\tau_{zy} + w\tau_{zz} - q_z]^T$$

where $E_t = \rho[e + \frac{1}{2}(u^2 + v^2 + w^2)]$.

The PNS equations are obtained from Eq. (1) by dropping the unsteady terms. The PNS equations expressed in a general nonorthogonal coordinate system (ξ, η, ζ) are given by

$$\boldsymbol{E}_\xi + \boldsymbol{F}_\eta + \boldsymbol{G}_\zeta = 0 \tag{2}$$

The $\boldsymbol{E}$ vector is frequently split for PNS applications using the Vigneron et al.[6] parameter ω. The parameter ω is given by

$$\omega = \min\left[1, \frac{\beta\gamma M_\xi^2}{1 + (\gamma - 1)M_\xi^2}\right]$$

where M_ξ is the local Mach number in the ξ direction and β is a safety factor that accounts for nonlinearities in the analysis. The $\boldsymbol{E}$ vector can then be written as

$$\boldsymbol{E} = \boldsymbol{E}^* + \boldsymbol{E}^p = A^*\boldsymbol{U} + A^p\boldsymbol{U} \tag{3}$$

where

$$\boldsymbol{E}^* = \frac{\xi_x}{J}\begin{bmatrix}\rho u\\ \rho u^2 + \omega p\\ \rho uv\\ \rho uw\\ (E_t + p)u\end{bmatrix} + \frac{\xi_y}{J}\begin{bmatrix}\rho v\\ \rho uv\\ \rho v^2 + \omega p\\ \rho vw\\ (E_t + p)v\end{bmatrix} + \frac{\xi_z}{J}\begin{bmatrix}\rho w\\ \rho uw\\ \rho vw\\ \rho w^2 + \omega p\\ (E_t + p)w\end{bmatrix}$$

$$\boldsymbol{E}^p = \frac{\xi_x}{J}\begin{bmatrix}0\\ (1-\omega)p\\ 0\\ 0\\ 0\end{bmatrix} + \frac{\xi_y}{J}\begin{bmatrix}0\\ 0\\ (1-\omega)p\\ 0\\ 0\end{bmatrix} + \frac{\xi_z}{J}\begin{bmatrix}0\\ 0\\ 0\\ (1-\omega)p\\ 0\end{bmatrix}$$

$$A^* = \frac{\partial \boldsymbol{E}^*}{\partial \boldsymbol{U}}, \qquad A^p = \frac{\partial \boldsymbol{E}^p}{\partial \boldsymbol{U}}$$

The $\boldsymbol{E}$ vector can also be split based on the eigenvalues[7] of the flux Jacobian $\partial\boldsymbol{E}/\partial\boldsymbol{U}$ so that

$$\boldsymbol{E} = \boldsymbol{E}^+ + \boldsymbol{E}^- = A^+\boldsymbol{U} + A^-\boldsymbol{U} \tag{4}$$

where $\boldsymbol{E}^+$ corresponds to the positive eigenvalues and $\boldsymbol{E}^-$ corresponds to the negative eigenvalues. The Jacobians A^+ and A^- can be readily computed using

$$A^+ = [T][\Lambda^+][T]^{-1}, \qquad A^- = [T][\Lambda^-][T]^{-1} \tag{5}$$

where $[T]$ is the matrix whose columns are the right eigenvectors of $\partial\boldsymbol{E}/\partial\boldsymbol{U}$ and $[\Lambda^+]$ and $[\Lambda^-]$ are diagonal matrices whose elements are the positive and negative eigenvalues, respectively.

Numerical Algorithms

The derivation of the numerical algorithms used in this study are now presented. The derivations given here provide details of the streamwise flux differencing and time discretizations using a finite volume formulation.

PNS Method

The PNS method[16–18] can be used to compute accurately supersonic viscous flows that contain no separated regions and produce negligible upstream influence. The PNS equations expressed in a general nonorthogonal coordinate system are given by Eq. (2):

$$\boldsymbol{E}_\xi + \boldsymbol{F}_\eta + \boldsymbol{G}_\zeta = 0$$

where the $\boldsymbol{E}$ vector is split using the Vigneron et al.[6] parameter:

$$\boldsymbol{E} = \boldsymbol{E}^* + \boldsymbol{E}^p \tag{6}$$

The streamwise flux vectors $\boldsymbol{E}^*$ and $\boldsymbol{E}^p$ are functions of the metrics, that is, geometry, designated by $d\boldsymbol{S}$ and the conservative flow

variables $\boldsymbol{U}$. It is convenient to represent the flux vectors at a given station ξ using the functional notation given by

$$\boldsymbol{E}^*_{i+1} = \boldsymbol{E}^*(d\boldsymbol{S}_{i+1}, \boldsymbol{U}_{i+1}), \qquad \boldsymbol{E}^p_{i+1} = \boldsymbol{E}^p(d\boldsymbol{S}_{i+1}, \boldsymbol{U}_{i+1}) \quad (7)$$

where the subscript $i+1$ denotes the spatial index (in the ξ direction) where the solution is currently being computed. The first-order accurate expression for the streamwise gradient can then be written as

$$\left(\frac{\partial \boldsymbol{E}}{\partial \xi}\right)_{i+1} = \frac{1}{\Delta\xi}\left[\left(\boldsymbol{E}^*_{i+1} - \boldsymbol{E}^*_i\right) + \left(\boldsymbol{E}^p_{i+1} - \boldsymbol{E}^p_i\right)\right] \quad (8)$$

The vectors $\boldsymbol{E}^*_{i+1}$ and $\boldsymbol{E}^p_{i+1}$ are then spatially linearized in the following manner:

$$\boldsymbol{E}^*_{i+1} = \boldsymbol{E}^*(d\boldsymbol{S}_{i+1}, \boldsymbol{U}_i) + \frac{\partial \boldsymbol{E}^*(d\boldsymbol{S}_{i+1}, \boldsymbol{U}_i)}{\partial \boldsymbol{U}}(\boldsymbol{U}_{i+1} - \boldsymbol{U}_i)$$

$$\boldsymbol{E}^p_{i+1} = \boldsymbol{E}^p(d\boldsymbol{S}_{i+1}, \boldsymbol{U}_i) + \frac{\partial \boldsymbol{E}^p(d\boldsymbol{S}_{i+1}, \boldsymbol{U}_i)}{\partial \boldsymbol{U}}(\boldsymbol{U}_{i+1} - \boldsymbol{U}_i) \quad (9)$$

where the Jacobians can be represented by

$$A^*(d\boldsymbol{S}_{i+1}, \boldsymbol{U}_i) = \frac{\partial \boldsymbol{E}^*(d\boldsymbol{S}_{i+1}, \boldsymbol{U}_i)}{\partial \boldsymbol{U}}$$

$$A^p(d\boldsymbol{S}_{i+1}, \boldsymbol{U}_i) = \frac{\partial \boldsymbol{E}^p(d\boldsymbol{S}_{i+1}, \boldsymbol{U}_i)}{\partial \boldsymbol{U}}$$

After the preceding linearizations and Jacobians are substituted into Eq. (8), the expression for the streamwise gradient becomes

$$\left(\frac{\partial \boldsymbol{E}}{\partial \xi}\right)_{i+1} = \frac{1}{\Delta\xi}\Big[\boldsymbol{E}^*(d\boldsymbol{S}_{i+1}, \boldsymbol{U}_i) + A^*(d\boldsymbol{S}_{i+1}, \boldsymbol{U}_i)\Delta\boldsymbol{U}_i$$

$$- \boldsymbol{E}^*_i + \boldsymbol{E}^p(d\boldsymbol{S}_{i+1}, \boldsymbol{U}_i) + A^p(d\boldsymbol{S}_{i+1}, \boldsymbol{U}_i) - \boldsymbol{E}^p_i\Big] \quad (10)$$

where $\Delta\boldsymbol{U}_i = \boldsymbol{U}_{i+1} - \boldsymbol{U}_i$. When the identities

$$\boldsymbol{E}^*_i = A^*_i\boldsymbol{U}_i, \qquad \boldsymbol{E}^*(d\boldsymbol{S}_{i+1}, \boldsymbol{U}_i) = A^*(d\boldsymbol{S}_{i+1}, \boldsymbol{U}_i)\boldsymbol{U}_i$$

$$\boldsymbol{E}^p_i = A^p_i\boldsymbol{U}_i, \qquad \boldsymbol{E}^p(d\boldsymbol{S}_{i+1}, \boldsymbol{U}_i) = A^p(d\boldsymbol{S}_{i+1}, \boldsymbol{U}_i)\boldsymbol{U}_i \quad (11)$$

are substituted into Eq. (10) and terms are simplified, the expression for the streamwise gradient becomes

$$\left(\frac{\partial \boldsymbol{E}}{\partial \xi}\right)_{i+1} = \frac{1}{\Delta\xi}\Big\{\Big[A^*(d\boldsymbol{S}_{i+1}, \boldsymbol{U}_i) + A^p(d\boldsymbol{S}_{i+1}, \boldsymbol{U}_i)\Big]\Delta\boldsymbol{U}_i$$

$$+ [A(d\boldsymbol{S}_{i+1}, \boldsymbol{U}_i) - A_i]\boldsymbol{U}_i\Big\} \quad (12)$$

where

$$A = A^* + A^p \quad (13)$$

When the PNS algorithm is utilized, the term $A^p(d\boldsymbol{S}_{i+1}, \boldsymbol{U}_i)\Delta\boldsymbol{U}_i$ is omitted to prevent departure solutions. The final discretized form of the PNS equations is obtained by substituting Eq. (12) into Eq. (2) along with the linearized expressions for the fluxes in the normal, η, and transverse, ζ, directions. The final expression becomes

$$\left\{\frac{1}{\Delta\xi}\Big[A^*(d\boldsymbol{S}_{i+1}, \boldsymbol{U}_i)\Big] + \frac{\partial}{\partial\eta}\left[\frac{\partial \boldsymbol{F}\left(d\boldsymbol{S}_{i+\frac{1}{2}}, \boldsymbol{U}_i\right)}{\partial \boldsymbol{U}}\right]\right.$$

$$\left. + \frac{\partial}{\partial\zeta}\left[\frac{\partial \boldsymbol{G}\left(d\boldsymbol{S}_{i+\frac{1}{2}}, \boldsymbol{U}_i\right)}{\partial \boldsymbol{U}}\right]\right\}\Delta\boldsymbol{U}_i = \text{RHS} \quad (14)$$

where the right-hand side (RHS) is

$$\text{RHS} = -\frac{1}{\Delta\xi}\{[A(d\boldsymbol{S}_{i+1}, \boldsymbol{U}_i) - A_i]\boldsymbol{U}_i\}$$

$$- \frac{\partial \boldsymbol{F}\left(d\boldsymbol{S}_{i+\frac{1}{2}}, \boldsymbol{U}_i\right)}{\partial\eta} - \frac{\partial \boldsymbol{G}\left(d\boldsymbol{S}_{i+\frac{1}{2}}, U_i\right)}{\partial\zeta}$$

Further details of this formulation used in NASA's UPS code may be found in Refs. 16 and 17.

FBIPNS Method

Forward Sweep

The FBIPNS approach utilizes Steger–Warming[7] flux splitting in the streamwise direction to account for upstream effects. In this section, the superscript $n+1$ will be used to denote the iteration level at which the forward-sweep solution is currently being computed. The flux vectors are evaluated at $(i+1, n+1)$ and the unsteady term is discretized using a first-order difference so that Eq. (1) becomes

$$\frac{1}{J\Delta t}\left(\boldsymbol{U}^{n+1}_{i+1} - \boldsymbol{U}^n_{i+1}\right) + \left(\frac{\partial \boldsymbol{E}}{\partial \xi}\right)^{n+1}_{i+1} + \left(\frac{\partial \boldsymbol{F}}{\partial \eta}\right)^{n+1}_{i+1} + \left(\frac{\partial \boldsymbol{G}}{\partial \zeta}\right)^{n+1}_{i+1} = 0 \quad (15)$$

By the addition and subtraction of $(1/J\Delta t)\boldsymbol{U}^{n+1}_i$, Eq. (15) can be written as

$$\frac{1}{J\Delta t}\left(\Delta\boldsymbol{U}^{n+1}_i\right) + \left(\frac{\partial \boldsymbol{E}}{\partial \xi}\right)^{n+1}_{i+1} + \left(\frac{\partial \boldsymbol{F}}{\partial \eta}\right)^{n+1}_{i+1} + \left(\frac{\partial \boldsymbol{G}}{\partial \zeta}\right)^{n+1}_{i+1}$$

$$= -\frac{1}{J\Delta t}\left(\boldsymbol{U}^{n+1}_i - \boldsymbol{U}^n_{i+1}\right) \quad (16)$$

The $\boldsymbol{E}$ vector is now split based on the eigenvalues[7] of $\partial\boldsymbol{E}/\partial\boldsymbol{U}$ so that

$$\boldsymbol{E} = \boldsymbol{E}^+ + \boldsymbol{E}^- = A^+\boldsymbol{U} + A^-\boldsymbol{U} \quad (17)$$

The streamwise gradient of $\boldsymbol{E}$ is then discretized using a backward difference for $\boldsymbol{E}^+$ and a forward difference for $\boldsymbol{E}^-$. Thus,

$$\left(\frac{\partial \boldsymbol{E}}{\partial \xi}\right)^{n+1}_{i+1} = \left(\frac{\partial \boldsymbol{E}^+}{\partial \xi}\right)^{n+1}_{i+1} + \left(\frac{\partial \boldsymbol{E}^-}{\partial \xi}\right)^{n+1}_{i+1}$$

$$= \frac{1}{\Delta\xi}\Big\{\left[(\boldsymbol{E}^+)^{n+1}_{i+1} - (\boldsymbol{E}^+)^{n+1}_i\right]$$

$$+ \left[\boldsymbol{E}^-\left(d\boldsymbol{S}_{i+1}, \boldsymbol{U}^n_{i+2}\right) - \boldsymbol{E}^-\left(d\boldsymbol{S}_i, \boldsymbol{U}^{n+1}_{i+1}\right)\right]\Big\} \quad (18)$$

The fluxes can be linearized in the following manner:

$$(\boldsymbol{E}^+)^{n+1}_{i+1} = \boldsymbol{E}^+\left(d\boldsymbol{S}_{i+1}, \boldsymbol{U}^{n+1}_i\right) + A^+\left(d\boldsymbol{S}_{i+1}, \boldsymbol{U}^{n+1}_i\right)\Delta\boldsymbol{U}^{n+1}_i$$

$$= A^+\left(d\boldsymbol{S}_{i+1}, \boldsymbol{U}^{n+1}_i\right)\boldsymbol{U}^{n+1}_i + A^+\left(d\boldsymbol{S}_{i+1}, \boldsymbol{U}^{n+1}_i\right)\Delta\boldsymbol{U}^{n+1}_i \quad (19)$$

$$(\boldsymbol{E}^+)^{n+1}_i = (A^+)^{n+1}_i\boldsymbol{U}^{n+1}_i \quad (20)$$

$$\boldsymbol{E}^-\left(d\boldsymbol{S}_{i+1}, \boldsymbol{U}^n_{i+2}\right) = \boldsymbol{E}^-\left(d\boldsymbol{S}_{i+1}, \boldsymbol{U}^{n+1}_i\right) + (A^-)^n_{i+1}\left(\boldsymbol{U}^n_{i+2} - \boldsymbol{U}^{n+1}_i\right)$$

$$= A^-\left(d\boldsymbol{S}_{i+1}, \boldsymbol{U}^{n+1}_i\right)\boldsymbol{U}^{n+1}_i + (A^-)^n_{i+1}\left(\boldsymbol{U}^n_{i+2} - \boldsymbol{U}^{n+1}_i\right) \quad (21)$$

$$\boldsymbol{E}^-\left(d\boldsymbol{S}_i, \boldsymbol{U}^{n+1}_{i+1}\right) = \boldsymbol{E}^-\left(d\boldsymbol{S}_i, \boldsymbol{U}^{n+1}_i\right) + (A^-)^{n+1}_i\Delta\boldsymbol{U}^{n+1}_i$$

$$= (A^-)^{n+1}_i\boldsymbol{U}^{n+1}_i + (A^-)^{n+1}_i\Delta\boldsymbol{U}^{n+1}_i \quad (22)$$

where $\Delta\boldsymbol{U}^{n+1}_i = \boldsymbol{U}^{n+1}_{i+1} - \boldsymbol{U}^{n+1}_i$.

The final discretized form of the equations used in the forward sweep of the FBIPNS approach is obtained by substituting Eq. (18)

into Eq. (16) and using linearizations as shown in Eqs. (19–22). The final expression becomes

$$\left\{\frac{1}{J\Delta t}+\frac{1}{\Delta\xi}\left[A^{+}(d\boldsymbol{S}_{i+1},\boldsymbol{U}_i)-A_i^{-}\right]^{n+1}\right.$$
$$+\frac{\partial}{\partial\eta}\left[\frac{\partial\boldsymbol{F}\left(d\boldsymbol{S}_{i+\frac{1}{2}},\boldsymbol{U}_i^{n+1}\right)}{\partial\boldsymbol{U}}\right]$$
$$\left.+\frac{\partial}{\partial\zeta}\left[\frac{\partial\boldsymbol{G}\left(d\boldsymbol{S}_{i+\frac{1}{2}},\boldsymbol{U}_i^{n+1}\right)}{\partial\boldsymbol{U}}\right]\right\}\Delta\boldsymbol{U}_i^{n+1}$$
$$=-\frac{1}{\Delta\xi}\left\{\left[A(d\boldsymbol{S}_{i+1},\boldsymbol{U}_i)-A_i\right]^{n+1}\boldsymbol{U}_i^{n+1}\right.$$
$$\left.+(A^{-})_{i+1}^{n}\left[\boldsymbol{U}_{i+2}^{n}-\boldsymbol{U}_i^{n+1}\right]\right\}-\frac{\partial\boldsymbol{F}\left(d\boldsymbol{S}_{i+\frac{1}{2}},\boldsymbol{U}_i^{n+1}\right)}{\partial\eta}$$
$$-\frac{\partial\boldsymbol{G}\left(d\boldsymbol{S}_{i+\frac{1}{2}},\boldsymbol{U}_i^{n+1}\right)}{\partial\zeta}-\frac{1}{J\Delta t}\left(\boldsymbol{U}_i^{n+1}-U_{i+1}^{n}\right)\tag{23}$$

This algorithm for the forward sweep is identical to the TIPNS algorithm and is first-order accurate in the streamwise direction and second-order accurate in the crossflow plane. A linear stability analysis of this differencing indicates that the first-order method is unconditionally stable for all time steps. This permits an infinite time step to be used (in most cases), which drops out the unsteady terms in Eq. (23). The time terms were omitted in all of the present calculations. The streamwise differencing can be extended to second-order accuracy as shown in Ref. 4. However, the second-order method does require a finite time step to ensure stability and also requires substantially more computer time because of the additional terms.

Backward Sweep

When marched from downstream to upstream stations, the solution at the $i-1$ station is computed with the solutions at the i and $i+1$ stations known. This backward sweep can be implemented, in a discrete form, as an exact mirror image of the forward-sweep algorithm explained in the preceding section. Thus, Eq. (1) can be discretized as

$$\frac{1}{J\Delta t}\left(\boldsymbol{U}_{i-1}^{n+1}-\boldsymbol{U}_{i-1}^{n}\right)+\left(\frac{\partial\boldsymbol{E}}{\partial\xi}\right)_{i-1}^{n+1}+\left(\frac{\partial\boldsymbol{F}}{\partial\eta}\right)_{i-1}^{n+1}+\left(\frac{\partial\boldsymbol{G}}{\partial\zeta}\right)_{i-1}^{n+1}=0\tag{24}$$

which, after adding and subtracting $(1/J\Delta t)\boldsymbol{U}_i^{n+1}$, becomes

$$\frac{1}{J\Delta t}\left(\Delta^{-}\boldsymbol{U}_i^{n+1}\right)+\left(\frac{\partial\boldsymbol{E}}{\partial\xi}\right)_{i-1}^{n+1}+\left(\frac{\partial\boldsymbol{F}}{\partial\eta}\right)_{i-1}^{n+1}+\left(\frac{\partial\boldsymbol{G}}{\partial\zeta}\right)_{i-1}^{n+1}$$
$$=-\frac{1}{J\Delta t}\left(\boldsymbol{U}_i^{n+1}-\boldsymbol{U}_{i-1}^{n}\right)\tag{25}$$

where $\Delta^{-}\boldsymbol{U}_i^{n+1}=\boldsymbol{U}_{i-1}^{n+1}-\boldsymbol{U}_i^{n+1}$.

The discretization of the streamwise flux $\boldsymbol{E}$ employs Steger–Warming[7] splitting and similar linearizations as used in the forward sweep:

$$\left(\frac{\partial\boldsymbol{E}}{\partial\xi}\right)_{i-1}^{n+1}=\left(\frac{\partial\boldsymbol{E}^{+}}{\partial\xi}\right)_{i-1}^{n+1}+\left(\frac{\partial\boldsymbol{E}^{-}}{\partial\xi}\right)_{i-1}^{n+1}$$
$$=\frac{1}{\Delta\xi}\left\{\left[(\boldsymbol{E}^{-})_i^{n+1}-(\boldsymbol{E}^{-})_{i-1}^{n+1}\right]\right.$$
$$\left.+\left[\boldsymbol{E}^{+}\left(d\boldsymbol{S}_i,\boldsymbol{U}_{i-1}^{n+1}\right)-\boldsymbol{E}^{+}\left(d\boldsymbol{S}_{i-1},\boldsymbol{U}_{i-2}^{n}\right)\right]\right\}\tag{26}$$

The fluxes can be linearized in the following manner:

$$(\boldsymbol{E}^{-})_i^{n+1}=(A^{-})_i^{n+1}\boldsymbol{U}_i^{n+1}\tag{27}$$

$$(\boldsymbol{E}^{-})_{i-1}^{n+1}=\boldsymbol{E}^{-}\left(d\boldsymbol{S}_{i-1},\boldsymbol{U}_i^{n+1}\right)+A^{-}\left(d\boldsymbol{S}_{i-1},\boldsymbol{U}_i^{n+1}\right)\Delta^{-}\boldsymbol{U}_i^{n+1}$$
$$=A^{-}\left(d\boldsymbol{S}_{i-1},\boldsymbol{U}_i^{n+1}\right)U_i^{n+1}+A^{-}\left(d\boldsymbol{S}_{i-1},U_i^{n+1}\right)\Delta^{-}\boldsymbol{U}_i^{n+1}\tag{28}$$

$$\boldsymbol{E}^{+}\left(d\boldsymbol{S}_i,\boldsymbol{U}_{i-1}^{n+1}\right)=\boldsymbol{E}^{+}\left(d\boldsymbol{S}_i,\boldsymbol{U}_i^{n+1}\right)+(A^{+})_i^{n+1}\Delta^{-}U_i^{n+1}$$
$$=(A^{+})_i^{n+1}\boldsymbol{U}_i^{n+1}+(A^{+})_i^{n+1}\Delta^{-}\boldsymbol{U}_i^{n+1}\tag{29}$$

$$\boldsymbol{E}^{+}\left(d\boldsymbol{S}_{i-1},\boldsymbol{U}_{i-2}^{n}\right)=\boldsymbol{E}^{+}\left(d\boldsymbol{S}_{i-1},\boldsymbol{U}_i^{n+1}\right)+(A^{+})_{i-1}^{n}\left(\boldsymbol{U}_{i-2}^{n}-\boldsymbol{U}_i^{n+1}\right)$$
$$=A^{+}\left(d\boldsymbol{S}_{i-1},\boldsymbol{U}_i^{n+1}\right)\boldsymbol{U}_i^{n+1}+(A^{+})_{i-1}^{n}\left(\boldsymbol{U}_{i-2}^{n}-\boldsymbol{U}_i^{n+1}\right)\tag{30}$$

The final discretized equation for the backward sweep can be obtained by substituting Eq. (26) into Eq. (25) and using the linearizations given by Eqs. (27–30). The final expression becomes

$$\left\{\frac{1}{J\Delta t}+\frac{1}{\Delta\xi}\left[A_i^{+}-A^{-}(d\boldsymbol{S}_{i-1},\boldsymbol{U}_i)\right]^{n+1}\right.$$
$$+\frac{\partial}{\partial\eta}\left[\frac{\partial\boldsymbol{F}\left(d\boldsymbol{S}_{i+\frac{1}{2}},\boldsymbol{U}_i^{n+1}\right)}{\partial\boldsymbol{U}}\right]$$
$$\left.+\frac{\partial}{\partial\zeta}\left[\frac{\partial\boldsymbol{G}\left(d\boldsymbol{S}_{i+\frac{1}{2}},\boldsymbol{U}_i^{n+1}\right)}{\partial\boldsymbol{U}}\right]\right\}\Delta^{-}\boldsymbol{U}_i^{n+1}$$
$$=-\frac{1}{\Delta\xi}\left\{\left[A_i-A(d\boldsymbol{S}_{i-1},\boldsymbol{U}_i)\right]^{n+1}\boldsymbol{U}_i^{n+1}\right.$$
$$\left.-(A^{+})_{i-1}^{n}\left[\boldsymbol{U}_{i-2}^{n}-\boldsymbol{U}_i^{n+1}\right]\right\}-\frac{\partial\boldsymbol{F}\left(d\boldsymbol{S}_{i+\frac{1}{2}},\boldsymbol{U}_i^{n+1}\right)}{\partial\eta}$$
$$-\frac{\partial\boldsymbol{G}\left(d\boldsymbol{S}_{i+\frac{1}{2}},\boldsymbol{U}_i^{n+1}\right)}{\partial\zeta}-\frac{1}{J\Delta t}\left(\boldsymbol{U}_i^{n+1}-\boldsymbol{U}_{i-1}^{n}\right)\tag{31}$$

A byproduct of the forward/backward sweeping scheme is that the streamwise flux terms are treated in a more symmetric fashion. Both sweeps have uneven linearizations in the streamwise inviscid flux, but when combined in an alternating direction fashion, the truncation error introduced by either sweep tends to be counterbalanced by the subsequent sweep in a manner reminiscent of alternating direction explicit schemes.[18] Thus, the order of accuracy of the scheme approaches second order in the marching direction.

Solution Procedure

The computations were started at an x station located at or upstream of the leading edge of the body geometry. In the region surrounding the leading edge, upstream effects may be important, and the FBIPNS algorithm can be used. The regions where the FBIPNS algorithm is applied are referred to as iterated regions. The extent of these regions can be determined automatically for arbitrary two-dimensional/axisymmetric supersonic flowfields using procedures developed in Ref. 19. For the present flowfields, the extent of the iterated regions were determined heuristically. The PNS method is used to obtain the initial solution in these regions. The convergence of the multiple-sweep FBIPNS method is checked by monitoring the rms residual of $\boldsymbol{U}$ and the skin friction after every sweep. Once the solution in the initial iterated region is fully converged, the next portion of the flowfield is computed with the single-sweep PNS method. This continues until the beginning of the next iterated region. Hence, each computation is broken up into regions of single-sweep PNS and multiple-sweep FBIPNS as shown in Fig. 1.

In the iterated regions, one can either employ forward-sweep iteration, that is, TIPNS, or forward/backward alternating direction sweeps, that is, FBIPNS. In the latter case, the backward sweep can be started immediately after the initial PNS solution is obtained. The backward sweep is started from the last (most downstream) crossflow plane in the iterated region and proceeds until it reaches the upstream end of the region. The forward sweep can then be

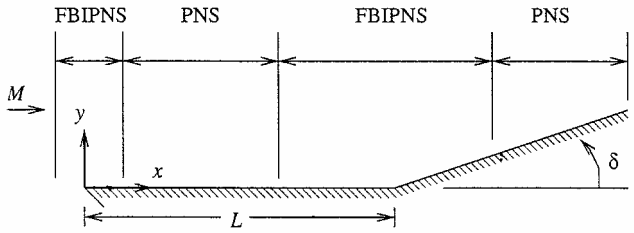

Fig. 1 Compression ramp configuration.

started from the first (most upstream) crossflow plane. At the end of the forward sweep, the backward sweep is again used to march upstream, and so on. For cases where obtaining the initial PNS solution is difficult, an impulsive start can be used. In this case, the forward-sweep TIPNS method is used to obtain the initial solution.

Streamwise Boundary Conditions

When a space-marching sweep reaches the last solution plane in the iterated region, an additional crossflow solution plane beyond the last plane must be filled using an appropriate boundary condition. The outflow boundary, that is, the downstreammost plane in the iterated region, is computed by using zeroth-order extrapolation from the previous interior station. At the inflow boundary, that is, the upstreammost plane in the region, the solution is fixed with initial values. Because the inflow boundary is placed far enough ahead of the region of upstream influence, this will not affect the accuracy of the solution.

UPS Code

The three-dimensional FBIPNS algorithm has been incorporated into NASA's upwind parabolized NS UPS code originally developed by Lawrence et al.[16,17] The UPS code solves the parabolized NS equations using a fully conservative, finite volume approach in a general nonorthogonal coordinate system. The UPS code was initially developed for perfect gas flows and uses an upwind, total variation diminishing method based on Roe's[20] approximate Riemann solver. During the past several years, the code has been modified to permit the accurate prediction of many types of supersonic and hypersonic flows including those in thermochemical nonequilibrium.[21]

Numerical Results

Test Case 1

This test case consists of the supersonic laminar flow over a flat plate followed by an 8-deg compression ramp. The geometry is shown in Fig. 1. The perfect gas flow conditions are as follows: $M_\infty = 3.0$, $Re = 1.68 \times 10^5$/m, $T_\infty = 216.67$ K, $T_w = 547.6$ K, and $\gamma = 1.4$.

The computations were started from freestream conditions at the leading edge and continued until an x station of 2.0 m was reached. The first grid point above the wall was located at 2×10^{-4} m. The height of the computational domain measured from the body surface was 1.25 m. The computations were initially obtained on a coarse grid consisting of 100 points in the normal y direction and 540 points in the streamwise x direction in the iterated region $0.40 < x < 2.0$ m. The iterated region was the same for both the TIPNS and FBIPNS calculations.

Figures 2 and 3 show comparisons of the streamwise variation of wall pressure and skin-friction coefficients, respectively. The wall pressures computed using the TIPNS and FBIPNS methods are in good agreement with the OVERFLOW[14] NS solution. The computed skin-friction results in Fig. 3 show some differences with the OVERFLOW results near the separated region. The differences are due to the streamwise accuracy of the present TIPNS and FBIPNS methods. The first-order-accurate TIPNS results show the most error, whereas the nearly second-order-accurate FBIPNS results agree much closer to the second-order OVERFLOW results. It has been shown in Ref. 4 that the second-order-accurate TIPNS method produces results that are identical to the OVERFLOW results. However, the second-order TIPNS method requires substantially more computer time than the present FBIPNS method.

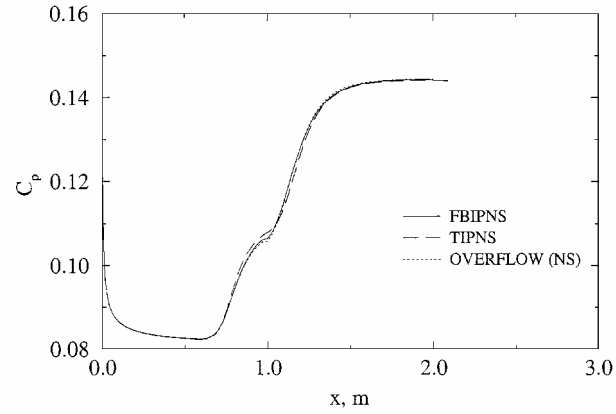

Fig. 2 Streamwise wall pressure coefficient variation for 8-deg compression ramp.

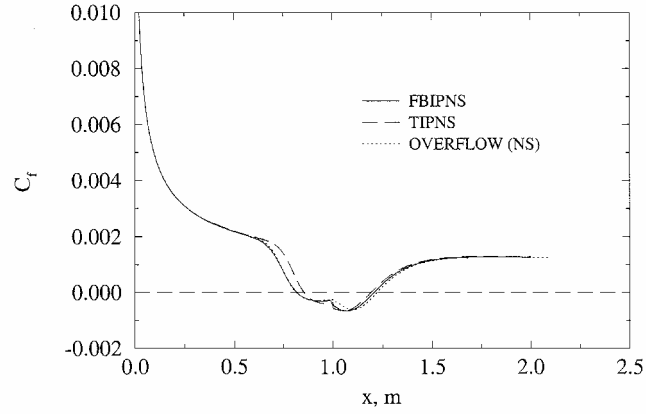

Fig. 3 Streamwise skin-friction variation for 8-deg compression ramp.

The forward-sweep TIPNS algorithm requires the use of a much finer grid in the streamwise direction to approach the OVERFLOW NS solution. In a typical PNS computation, the streamwise grid spacing is finer than that used in time-dependent NS solvers such as OVERFLOW. This is due in part to the first-order streamwise accuracy and also the stability constraint. However, the finer streamwise step size used in the PNS/TIPNS computations does not require significant increases in computer time due to the efficiency of the space-marching algorithm, and it has been shown in previous studies[3,4] that the overall computational time required for the forward-sweep TIPNS method is much faster than time-dependent NS computations. Further decreases in the computational time can be achieved by using the present forward/backward space-marching algorithm because a coarser grid can be used. For the present test case, the OVERFLOW NS calculation required 124 min of CPU time on a DEC Alphastation 500/400, the TIPNS calculation required 54 min, and the FBIPNS calculation required 26 min. These results were obtained using the same coarse grid for each calculation.

Figure 4 shows the convergence history of the forward-only, that is, TIPNS, and forward/backward, that is, FBIPNS, computations on the coarse grid. The FBIPNS algorithm clearly improves the convergence rate. Along with the earlier observation that the TIPNS algorithm requires finer streamwise grid spacing, this translates to a significant savings in computational time.

A grid refinement study (Fig. 5) indicates that further refinement of the coarse grid did not change the FBIPNS results. However, the TIPNS results do improve somewhat as the grid is refined and approach the coarse grid FBIPNS results.[4] The first grid point off the wall for each of the three grids is located at 2.0×10^{-4}, 1.0×10^{-4}, and 0.5×10^{-4} m, respectively. In addition, a study was conducted to determine the approximate accuracy of the FBIPNS scheme in the marching direction x. When two Richardson extrapolations are used in conjunction with the computed results from three grids (with

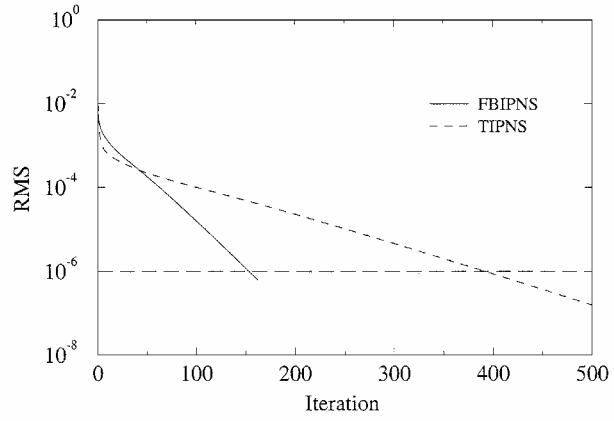

Fig. 4 Convergence history for 8-deg compression ramp computations.

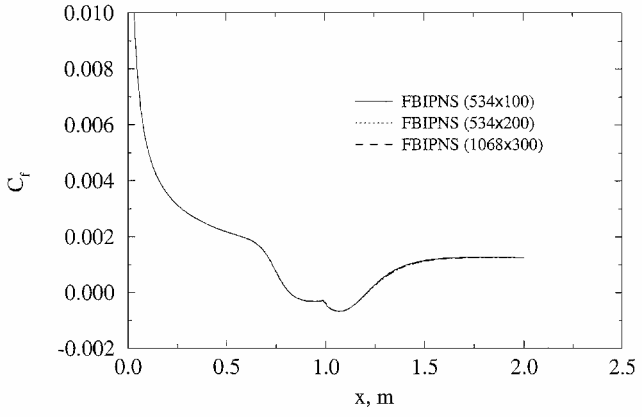

Fig. 5 Grid independence study for 8-deg compression ramp case.

grid spacings Δ, 2Δ, 4Δ), it can be shown that the order of accuracy n of a numerical scheme is given by

$$n = \ell n\left[\frac{Q(2\Delta) - Q(4\Delta)}{Q(\Delta) - Q(2\Delta)}\right] \Big/ \ell n(2) \tag{32}$$

where $Q(\Delta)$, $Q(2\Delta)$, and $Q(4\Delta)$ represent the numerical solutions of a flow quantity computed at a given point with grid spacings of Δ, 2Δ, and 4Δ, respectively. For the present study, the wall pressures were computed using the grids (267×200), (534×200), and (1068×200) and were compared at common grid points. The value of n was then computed at each grid point using Eq. (32). The average value for n (the order of accuracy in the x direction) was found to be 1.90, which approaches second-order accuracy ($n = 2$) as stated, earlier.

Test Case 2

The hypersonic laminar flow over a hollow-cylinder-flare test case has been computed using the TIPNS and FBIPNS algorithms. The numerical test case corresponds to Holden's[13] experimental test case run 11. The experimental tests were conducted using a nitrogen freestream to remove the effects of flowfield chemistry. Two different geometric configurations were tested including an extended flare geometry (Fig. 6) and a truncated flare geometry. Only the external flow over the extended flare geometry was computed in the present study. Preliminary calculations made during this study indicate that the internal flow has no effect on the external flow.

The geometric dimensions of the body are given in Ref. 13. The following flow parameters were used in the perfect gas numerical calculations: $p_\infty = 19.37$ N/m^2, $\rho_\infty = 5.065 \times 10^{-4}$ kg/m^3, $T_\infty = 128.9$ K, $V_\infty = 2609$ m/s, $M_\infty = 11.28$, $Re_\infty = 1.5196 \times 10^5$/m, $Pr = 0.71$, $\gamma = 1.4$, $T_w = 297.2$ K, and $R = 296.7$ m^2/s^{2}K. The freestream density, temperature, and velocity match exactly the test conditions given by Holden.[13] The coefficient of viscosity (in newton seconds per square meter) was computed using Sutherland's equation for nitrogen

$$\mu = 1.3998 \times 10^{-6}\left[T^{\frac{3}{2}} \Big/ (T + 106.67)\right]$$

and the coefficient of thermal conductivity was computed by assuming a constant Prandtl number.

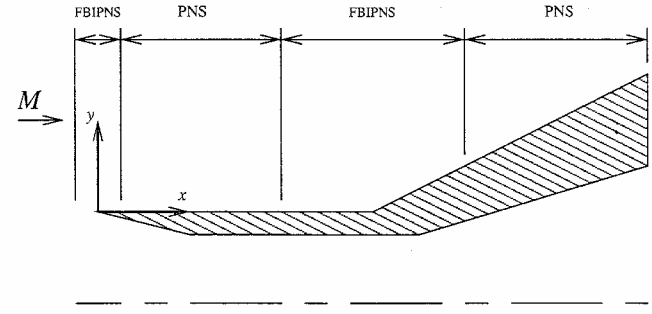

Fig. 6 Hollow-cylinder-flare configuration.

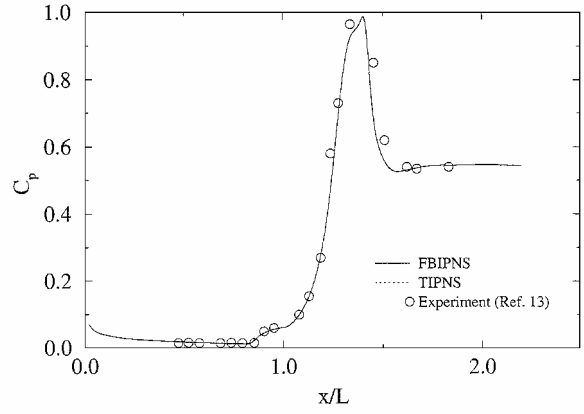

Fig. 7 Wall pressure coefficient variation for test case 2.

The present test case was computed using several different grids to ensure grid independence. The most refined grid, which was used to obtain the present results, is now described. The first iterated region (which straddles the leading edge) starts at $x/L = -0.01$ and ends at $x/L = 0.05$. The streamwise grid spacing Δx is 1.0×10^{-5} m at both inflow and outflow boundaries of the region and is clustered using linear interpolation centered at the leading edge, $x/L = 0$, at which $\Delta x = 1.0 \times 10^{-7}$ m. The normal grid spacing at the wall Δy is 1.0×10^{-7} m at $x/L = 0$ and 1.0×10^{-6} m at $x/L = 0.05$. The normal grid spacing at the wall is kept at 1.0×10^{-6} m for the remainder of the flowfield. Within both PNS regions (Fig. 6) the streamwise spacing is $\Delta x = 1.0 \times 10^{-4}$ m. The grid for the second iterated region, which starts at $x/L = 0.68$ and ends at $x/L = 1.63$, contains 3800 points in the streamwise direction and 300 in the normal direction. The streamwise grid spacing is 4.0×10^{-5} m. The regions of iteration were the same for both the TIPNS and FBIPNS calculations.

Figures 7–9 show the results for the wall pressure coefficient, the Stanton number, and the skin-friction coefficient variations in the streamwise direction. The coefficients are computed as follows:

$$C_p = (p - p_\infty) \Big/ \tfrac{1}{2}\rho_\infty V_\infty^2, \qquad St = q_w \Big/ \tfrac{1}{2}\rho_\infty V_\infty^3$$

$$C_f = \tau_w \Big/ \tfrac{1}{2}\rho_\infty V_\infty^2$$

where q_w is the heat transfer rate at the wall and τ_w is the wall skin friction. A small separated region is induced by the flare, as seen in Fig. 9. The TIPNS and FBIPNS methods produce nearly identical results for this case because of the small separated region. Both methods agree closely with the experimental data of Holden.[13] In fact, in a "blind" code validation study,[22] the present results[9] agreed closer to the experimental data than did the other four numerical (three NS and one direct simulation Monte Carlo) calculations. The

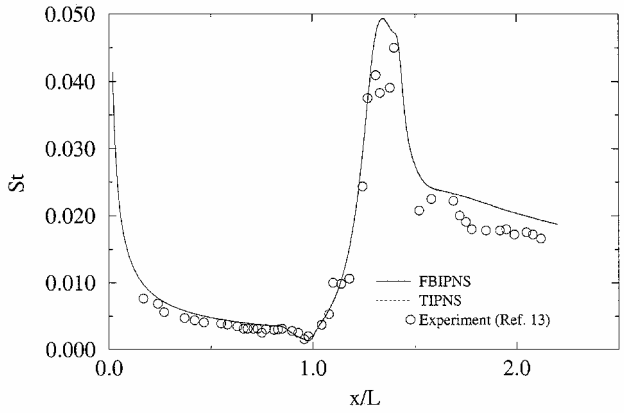

Fig. 8 Heat transfer coefficient variation for test case 2.

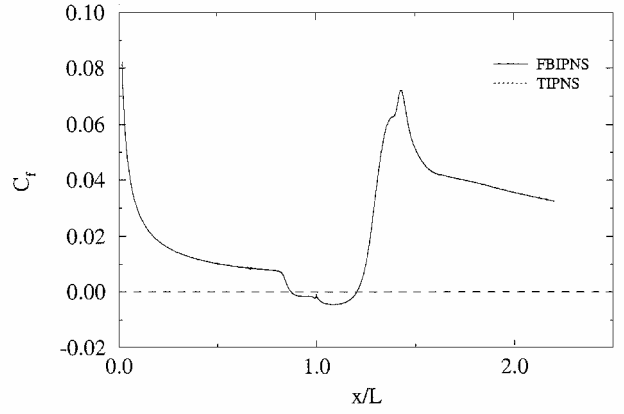

Fig. 9 Skin-friction coefficient variation for test case 2.

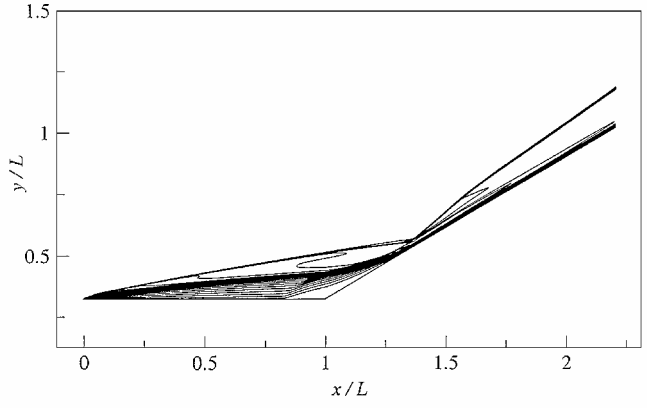

Fig. 10 Mach number contours for test case 2.

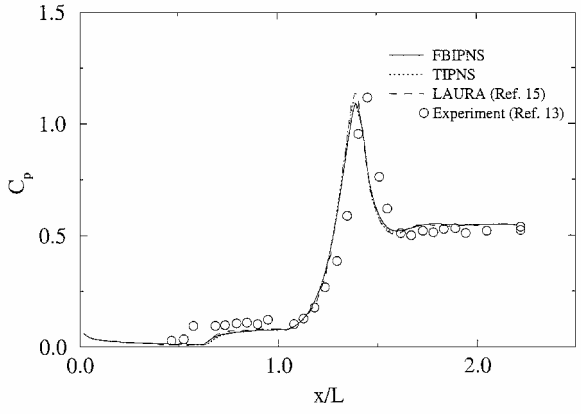

Fig. 11 Wall pressure coefficient variation for test case 3.

accuracy of the experimental measurements[13] are +3% for wall pressure and +5% for heat transfer. A plot of the computed Mach number contours is shown in Fig. 10.

Test Case 3

This case corresponds to Holden's experimental run 14. The geometric configuration is identical to test case 2. The flow parameters are as follows: $p_\infty = 36.76$ N/m^2, $\rho_\infty = 7.9369 \times 10^{-4}$ kg/m^3, $T_\infty = 156.1$ K, $V_\infty = 2432$ m/s, $M_\infty = 9.55$, $Re_\infty = 1.8089 \times 10^5$/m, $Pr = 0.71$, $\gamma = 1.4$, $T_w = 295.6$ K, and $R = 296.7$ m^2/s^{2}K. As with the preceding case, the freestream density, temperature, and velocity match exactly the test conditions given by Holden.[13] The grid used in the present calculation is identical to the preceding case except that the second iterated region extends from $x/L = 0.49$ to 1.63.

Figures 11–13 show the wall pressure coefficient, the Stanton number, and the skin-friction variations in the streamwise direction. As is evident in Fig. 13, the separated region for this case

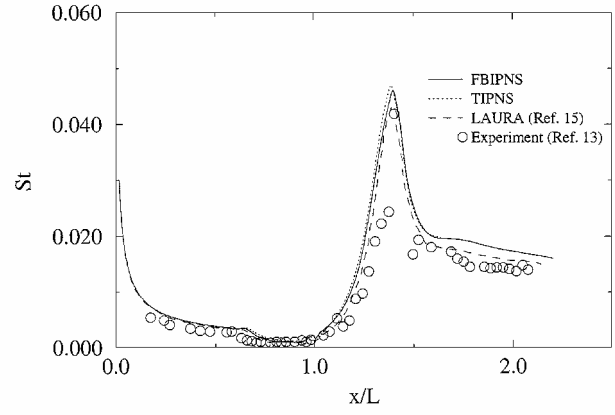

Fig. 12 Heat transfer coefficient variation for test case 3.

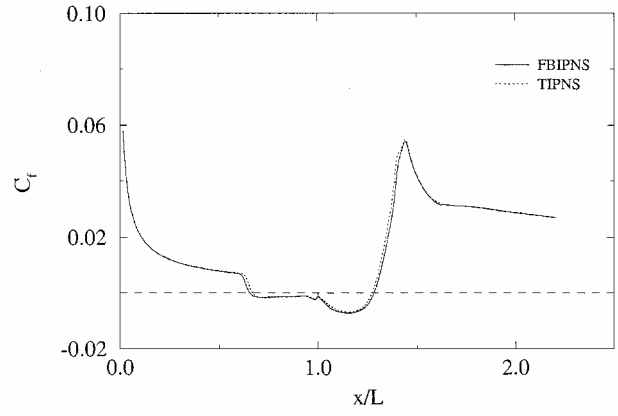

Fig. 13 Skin-friction coefficient variation for test case 3.

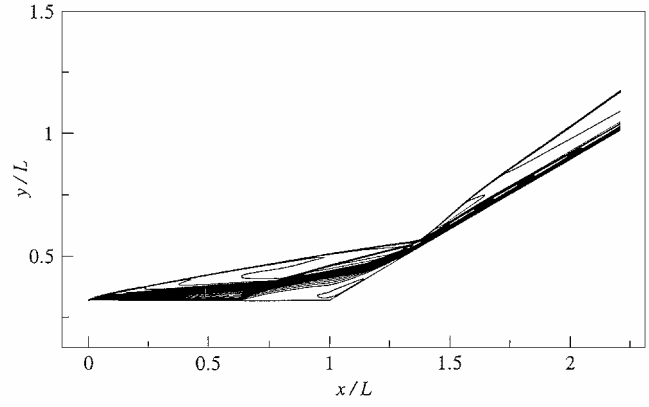

Fig. 14 Mach number contours for test case 3.

is significantly larger than the preceding case. The FBIPNS results show a slightly larger separated region than that given by the TIPNS calculation. Both FBIPNS and TIPNS results predict a smaller separation region than is indicated by Holden's[13] experimental results. For comparison, the NS results computed by Gnoffo[15] using the LAURA code are also included in Figs. 11 and 12. Gnoffo's numerical results are very similar to the FBIPNS/TIPNS results. The computed Mach number contours are shown in Fig. 14.

Conclusions

A forward/backward sweeping FBIPNS algorithm has been developed to reduce significantly the computer time and storage required to compute supersonic/hypersonic flowfields with embedded separated regions. The new algorithm has been tested by applying it to three separated flow cases, and the numerical results are in excellent agreement with full NS computations and experimental data. The three test cases include large regions of upstream influence that cover a substantial portion of the computational domain. For these cases, the present algorithm will not be substantially faster than an NS solver. On the other hand, for typical aerospace vehicle flowfields, where the separated regions encompass a small fraction of the total computational domain, the present approach will significantly reduce the computer time and storage in comparison to an NS calculation.

References

[1]Miller, J. H., Tannehill, J. C., and Lawrence, S. L., "Computation of Supersonic Flows with Embedded Separated Regions Using an Efficient PNS Algorithm," AIAA Paper 97-1942, June 1997.

[2]Miller, J. H., Tannehill, J. C., and Lawrence, S. L., "PNS Algorithm for Solving Supersonic Flows with Upstream Influences," AIAA Paper 98-0226, Jan. 1998.

[3]Miller, J. H., Tannehill, J. C., and Lawrence, S. L., "Parabolized Navier–Stokes Algorithm for Solving Supersonic Flows with Upstream Influences, *AIAA Journal*, Vol. 38, No. 10, 2000, pp. 1837–1845.

[4]Tannehill, J. C., Miller, J. H., and Lawrence, S. L., "Development of an Iterative PNS Code for Separated Flows," AIAA Paper 99-3361, June 1999.

[5]Tannehill, J. C., Miller, J. H., and Lawrence, S. L., "Iterative PNS Algorithms for Solving 3-D Supersonic Flows with Upstream Influences," AIAA Paper 2000-0821, Jan. 2000.

[6]Vigneron, Y. C., Rakich, J. V., and Tannehill, J. C., "Calculation of Supersonic Flow over Delta Wings with Sharp Subsonic Leading Edges," AIAA Paper 78-1137, July 1978.

[7]Steger, J. L., and Warming, R. F., "Flux Vector Splitting of the Inviscid Gasdynamic Equations with Application to Finite-Difference Methods," *Journal of Computational Physics*, Vol. 40, No. 2, 1981, pp. 263–293.

[8]Lahiri, S. K., and Tannehill, J. C., "Computation of Hypersonic Separated Flows over Compression Ramps Using the TIPNS Algorithm," AIAA Paper 2000-2599, June 2000.

[9]Kato, H., and Tannehiill, J. C., "Computation of Hypersonic Laminar Separated Flows Using an Iterated PNS Algorithm," AIAA Paper 2001-1028, Jan. 2001.

[10]Barnett, M., and Davis, R. T., "Calculation of Supersonic Flows with Strong Viscous–Inviscid Interaction," *AIAA Journal*, Vol. 24, No. 12, 1986, pp. 1949–1955.

[11]Stookesberry, D. C., and Tannehill, J. C., "Computation of Separated Flow Using the Space-Marching Conservative Supra-Characteristics Method," *AIAA Journal*, Vol. 25, No. 8, 1987, pp. 1063–1070.

[12]Chang, C.-L., and Merkle, C. L., "The Relation Between Flux Vector Splitting and Parabolized Schemes," *Journal of Computational Physics*, Vol. 80, No. 2, 1989, pp. 344–361.

[13]Holden, M. S., "Experimental Database from CUBRC Studies in Hypersonic Laminar and Turbulent Interacting Flows Including Flowfield Chemistry," Calspan—Univ. at Buffalo Research Center, Buffalo, NY, Sept. 2000.

[14]Buning, P. G., Jespersen, D. C., and Pulliam, T. H., "OVERFLOW Manual," Ver. 1.7v, NASA Ames Research Center, Moffett Field, CA, June 1997.

[15]Gnoffo, P. A., "CFD Validation Studies for Hypersonic Flow Prediction," AIAA Paper 2001-1025, Jan. 2001.

[16]Lawrence, S. L., Tannehill, J. C., and Chaussee, D. S., "Upwind Algorithm for the Parabolized Navier–Stokes Equations," *AIAA Journal*, Vol. 27, No. 9, 1989, pp. 1175–1183.

[17]Lawrence, S. L., Chaussee, D. S., and Tannehill, J. C., "Application of an Upwind Algorithm to the Three-Dimensional Parabolized Navier–Stokes Equations, *AIAA Journal*, Vol. 28, No. 6, 1990, pp. 971, 972.

[18]Tannehill, J. C., Anderson, D. A., and Pletcher, R. H., *Computational Fluid Mechanics and Heat Transfer*, Taylor and Francis, Washington, DC, 1997.

[19]Miller, J. H., Tannehill, J. C., and Lawrence, S. L., "Application of the IPNS Algorithm to Complex Two-Dimensional Body Geometries," AIAA Paper 99-0546, Jan. 1999.

[20]Roe, P. L., "Approximate Riemann Solvers, Parameters, Vectors and Difference Schemes," *Journal of Computational Physics*, Vol. 43, No. 2, 1983, pp. 357–372.

[21]Miller, J. H., Tannehill, J. C., Lawrence, S. L., and Edwards, T. A., "Parabolized Navier–Stokes Code for Hypersonic Flows in Thermo-Chemical Equilibrium or Nonequilibrium," *Computers and Fluids*, Vol. 27, No. 2, 1998, pp. 199–215.

[22]Harvey, J. K., Holden, M. S., and Wadhams, T. P., "Code Validation Study of Laminar Shock/Boundary Layer and Shock/Shock Interactions in Hypersonic Flow," AIAA Paper 2001-1031, Jan. 2001.

B. Hassan
Associate Editor