Journal of Computational Physics **179,** 140–169 (2002)
doi:10.1006/jcph.2002.7048

# The Use of Domain Decomposition in Accelerating the Convergence of Quasihyperbolic Systems

Bernard Parent and Jean P. Sislian

*University of Toronto Institute for Aerospace Studies, 4925 Dufferin St., Downsview, Ontario M3H 5T6, Canada*
E-mail: bernard.parent@utoronto.ca, sislian@caius.utias.utoronto.ca

Received March 28, 2001; revised March 5, 2002

This paper proposes an alternate form of the active-domain method [K. Nakahashi and E. Saitoh, *AIAA J*. **35**, 1280 (1997)] that is applicable to streamwise separated flows. Named the “marching window,” the algorithm consists of performing pseudo-time iterations on a minimal width subdomain composed of a sequence of cross-stream planes of nodes. The upstream boundary of the subdomain is positioned such that all nodes upstream exhibit a residual smaller than the user-specified convergence threshold. The advancement of the downstream boundary follows the advancement of the upstream boundary, except in zones of significant streamwise ellipticity, where a streamwise ellipticity sensor ensures its continuous progress. Compared to the standard pseudo-time-marching approach, the marching window decreases the work required for convergence by up to 24 times for flows with little streamwise ellipticity and by up to eight times for flows with large streamwise separated regions. Storage is reduced by up to six times by not allocating memory to the nodes not included in the computational subdomain. The marching window satisfies the same convergence criterion as the standard pseudo-time-stepping methods, hence resulting in the same converged solution within the tolerance of the user-specified convergence threshold. The algorithm is not restricted to a discretization stencil and pseudo-time-stepping scheme in particular and is used here with the Yee–Roe scheme and block-implicit approximate factorization solving the Favre-averaged Navier–Stokes (FANS) equations closed by the Wilcox $k\omega$ turbulence model. The eigenstructure of the FANS equations is also presented. © 2002 Elsevier Science (USA)

*Key Words:* convergence acceleration; viscous hypersonic flow; space marching; pseudo-time stepping; domain decomposition; FANS; RANS.

## 1. INTRODUCTION

There is little doubt that the most efficient way to solve supersonic or hypersonic flow with no streamwise ellipticity is through a space-marching method, as numerous extremely efficient marching methods developed over the years can attest (see, for example, Refs. [1–5]). The Navier–Stokes equations at supersonic speeds do, however, exhibit some ellipticity in the marching direction through the streamwise viscous terms and the subsonic layer of the boundary layer, and it is necessary for a space-marching method to ignore these mechanisms by solving a reduced set of the original equations of motion, such as the parabolized Navier–Stokes equations (PNS). The PNS are defined here as the equation set obtained from the Navier–Stokes equations by neglecting all viscous terms in the streamwise direction and by modifying the streamwise momentum equation to prevent any pressure disturbance to travel upstream, using characteristics splitting or pressure splitting, as suggested by Vigneron *et al.* [1]. The applicability of the space-marching methods is limited to flows with negligible streamwise ellipticity, hence preventing their deployment to many practical flow fields.

The need to tackle streamwise ellipticity prompted the development of the "global iteration" space-marching methods, in which a sweep is performed several times on the entire computational domain to permit the upstream propagation of information (see Ref. [6] for a detailed review). Such are characterized, compared to the pseudo-time-marching schemes, by a smaller memory requirement due to the storage of temporary variables in one marching plane only and by an enhanced wave-propagation mechanism in the streamwise direction. The reduced Navier–Stokes (RNS) equations, which are derived from the Navier–Stokes equations by ignoring all streamwise diffusion terms but not altering the momentum convection terms, are usually solved in this manner, leading to fast convergence of subsonic/supersonic streamwise unseparated flows [7–9], and even to viscous/inviscid interactions creating streamwise separation [10, 11]. In a similar vein, Bardina [12] shows that significant reduction in work is achievable by the use of global marching sweeps to solve the full Navier–Stokes equations for high-speed flows. However, if not limited to some predetermined zones of the computational domain, the global iteration approach exhibits poor efficiency when solving large reverse-flow regions, as the number of sweeps can become excessive due to its dependence on the size of the separation bubble. Further, some computing might be inefficiently allocated to the nodes downstream of the separation bubble, *prior* to its convergence. These deficiencies can be remedied by using a space-marching scheme solving the PNS equations until an elliptic/reverse-flow region is encountered, then switching to a global iteration RNS method for the length of the elliptic region, iterating until convergence is reached, and pursuing with the marching PNS scheme (see Miller *et al.* [6], for instance). However, such a strategy forces the solution of the PNS equations in certain regions of the flow field, for which the PNS assumption might induce appreciable errors. The accuracy of the final solution is hence strongly dependent on the ability of the method to predict correctly which regions of the flow field can be accurately predicted with the PNS equations, and which regions require the use of the RNS equations.

Recently, a novel approach to solving inviscid supersonic flow with embedded subsonic regions has been proposed [13]. The method, named "active domain," consists of performing pseudo-time iterations on a small bandlike computational domain that advances in the streamwise direction every time the residual of the active domain near the upstream boundary falls below a user-defined threshold. Using sensors based on the streamwise components of the Mach number, the active-domain boundaries automatically surround any locally subsonic region on which sufficient iterations are performed to reach steady state. When the residual inside the subsonic region decreases below the user-defined threshold, the active domain advances past the subsonic region further downstream. By marching in the streamwise direction, the active domain results in up to a 10-fold decrease in work compared to standard pseudo-time-marching methods for several inviscid problems. However, the ability of the active domain to solve accurately a streamwise elliptic region is limited by the accuracy of the sensor responsible for the upstream movement of the upstream boundary of the active domain. Extension of the active-domain method to viscous flow is hampered by the difficulty of formulating a streamwise ellipticity sensor that captures all significant upstream propagating waves while restricting the size of the active domain to a minimum. Success has been reported in solving viscous flow without streamwise separation by maintaining an active-domain width equal to the height of the boundary layer [14]. However, to the authors' knowledge, the active-domain method has not yet been extended to streamwise separated flows.

This paper proposes an alternate form of the active-domain method, named the "marching-window" algorithm, which is applicable to streamwise separated flows. Similarly to the active domain, the marching window performs localized pseudo-time stepping on a subdomain composed of a sequence of cross-stream planes of nodes. The width of the marching window decreases to only a few planes in regions of quasihyperbolic flow and increases to the size of the streamwise-elliptic region when encountered. However, in contrast to the active-domain algorithm, the marching window is strictly a convergence acceleration technique, as it guarantees that the residual of all nodes will be below the user-defined threshold when convergence is attained. This is accomplished by keeping the residual upstream of the marching-window subdomain updated at all times, and by positioning the upstream boundary such that the residual of all nodes upstream is below the user-defined threshold. This results in an algorithm that captures all upstream propagating waves affecting the residual significantly. The upstream propagating waves can originate from (but are not necessarily limited to) large subsonic pockets, streamwise separation, streamwise viscous fluxes, or flux limiters in the streamwise convection flux derivative, for instance. Further, to enhance the performance of the algorithm, a sensor based on the Vigneron splitting [1] is developed to advance the downstream boundary when significant steamwise ellipticity is detected.

Several numerical experiments are presented, ranging from the inviscid solution of a supersonic inlet with a blunt leading edge to turbulent shock boundary layer interactions with considerable streamwise flow separation solved with the Favre-averaged Navier–Stokes (FANS) equations closed by the $k\omega$ turbulence model of Wilcox [15]. A time-accurate turbulent flow field using dual time stepping is also investigated. A comparison between the marching-window cycle, the active-domain cycle (for the inviscid case only), and the standard pseudo-time-marching cycle is made on the basis of CPU time, effective iterations, and storage.

## 2. GOVERNING EQUATIONS

The residual of the full Navier–Stokes equations can be expressed in generalized coordinates in tensor form for any number of dimensions as

$$R = \sum_{i=1}^{d}\left[\frac{\partial F_i}{\partial X_i} - \sum_{j=1}^{d}\frac{\partial}{\partial X_i}\left(K_{ij}\frac{\partial G}{\partial X_j}\right)\right] - S, \tag{1}$$

where the minimization of $R$ is sought and where $d$ refers to the number of dimensions. Due to the nonlinearity of the system of equations, a fictitious unsteady term $\partial Q/\partial\tau$ is necessary to obtain the right physical root from a given set of initial conditions, i.e.,

$$\frac{\partial Q}{\partial \tau} = -R. \tag{2}$$

Even though the marching-window method presented in this paper and the discretization and pseudo-time-stepping schemes are not linked to some governing equations in particular, this study focuses on the Favre-averaged Navier–Stokes equations (FANS) closed by the Wilcox two-equation $k\omega$ model [15]. The $k\omega$ turbulence model is chosen due to its capability of solving accurately a wide range of realistic flow fields, while maintaining a close resemblance in form to the classical NS equations and avoiding the use of extra low Reynolds number terms (present in the $k\varepsilon$ family of turbulence models, for instance). Aside from their inelegance, additional low Reynolds number terms are detrimental to the performance of the algorithm because they increase the complexity of the source terms, which results in the need for bigger meshes in order to attain grid convergence, and because they increase the stiffness of the governing equations, which often translates into increased convergence time. This is avoided by the $k\omega$ modeling, which induces a conservative variable $Q$, a convective flux $F_i$, and a diffusion term $G$ of

$$Q = \frac{1}{J}\begin{bmatrix} \rho \\ \rho v_1 \\ \vdots \\ \rho v_d \\ \rho E \\ \rho k \\ \rho\omega \end{bmatrix}, \quad F_i = \frac{1}{J}\begin{bmatrix} \rho V_i \\ \rho V_i v_1 + X_{i,1}P^\star \\ \vdots \\ \rho V_i v_d + X_{i,d}P^\star \\ \rho V_i H \\ \rho V_i k \\ \rho V_i \omega \end{bmatrix}, \quad \text{and} \quad G = \begin{bmatrix} 1 \\ v_1 \\ \vdots \\ v_d \\ T \\ k \\ \omega \end{bmatrix}, \tag{3}$$

where the notation $X_{i,j}$ stands for $\partial X_i/\partial x_j$ and $J$ is the metric Jacobian, both of which are obtainable in any dimension following the approach of Viviand [16] and Vinokur [17]. The total energy, enthalpy, and effective pressure include molecular and turbulent properties,

$$E = e + k + \frac{1}{2}\sum_{i=1}^{d} v_i^2, \quad P^\star = P + \frac{2}{3}\rho k, \quad \text{and} \quad H = E + \frac{P^\star}{\rho}, \tag{4}$$

where $e$ is the internal energy of the gas, $v_i$ the velocity component in the Cartesian $x_i$ direction, and $k$ the turbulence kinetic energy. Calorically perfect gas assumptions are used to determine the internal energy from the temperature while an ideal gas law is assumed in finding $P$ from $\rho$ and $T$.

Starting from the tensorial form of the governing equations in Cartesian coordinates, it can be shown that the diffusion matrix $K_{ij}$ corresponds in curvilinear coordinates to

$$K_{ij} = \frac{1}{J}\begin{bmatrix} 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & \mu^\star\beta_{ij}^{11} & \cdots & \mu^\star\beta_{ij}^{1d} & 0 & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ 0 & \mu^\star\beta_{ij}^{d1} & \cdots & \mu^\star\beta_{ij}^{dd} & 0 & 0 & 0 \\ 0 & \mu^\star\Sigma_k\beta_{ij}^{k1}\nu_k & \cdots & \mu^\star\Sigma_k\beta_{ij}^{kd}\nu_k & \kappa^\star\alpha_{ij} & \mu_k^\star\alpha_{ij} & 0 \\ 0 & 0 & \cdots & 0 & 0 & \mu_k^\star\alpha_{ij} & 0 \\ 0 & 0 & \cdots & 0 & 0 & 0 & \mu_\omega^\star\alpha_{ij} \end{bmatrix}, \tag{5}$$

where the diffusion coefficients include a molecular and turbulent contribution function of the eddy viscosity $\mu_t = 0.09\rho k/\omega$, molecular viscosity $\mu$, and molecular thermal conductivity $\kappa$; that is,

$$\mu^\star = \mu + \mu_t, \quad \kappa^\star = \kappa + C_P\frac{\mu_t}{0.9}, \quad \mu_k^\star = \mu + \frac{\mu_t}{2}, \quad \text{and} \quad \mu_\omega^\star = \mu + \frac{\mu_t}{2}, \tag{6}$$

while $\alpha$ and $\beta$ are a function of the metrics only.

$$\alpha_{ij} = \sum_{k=1}^{d} X_{i,k}X_{j,k} \quad \text{and} \quad \beta_{ij}^{mn} = \alpha_{ij}\delta_{mn}^{\text{Kr}} + X_{j,m}X_{i,n} - \frac{2}{3}X_{j,n}X_{i,m}, \tag{7}$$

where $\delta_{mn}^{\text{Kr}}$ stands for the Kronecker delta, which vanishes should $m$ and $n$ differ but assumes a value of unity otherwise. The source term $S$ is composed of the sum of the baseline Wilcox $k\omega$ source terms [15] and an unsteady source term,

$$S = \frac{1}{J}\begin{bmatrix} \vdots \\ 0 \\ P_k - \rho k\omega \\ \frac{\omega}{\tilde{k}}\left(\frac{5}{9}P_k - \frac{5}{6}\rho k\omega\right) \end{bmatrix} - \frac{\partial Q}{\partial t}, \tag{8}$$

where the turbulence kinetic energy production term $P_k$ can be shown to correspond to

$$P_k = \sum_{i=1}^{d}\sum_{j=1}^{d}\left(-\frac{2}{3}\rho k X_{i,j}\frac{\partial \nu_j}{\partial X_i} + \sum_{m=1}^{d}\sum_{n=1}^{d}\mu^\star\beta_{ij}^{mn}\frac{\partial \nu_m}{\partial X_i}\frac{\partial \nu_n}{\partial X_j}\right). \tag{9}$$

It is noted that in the Wilcox $k\omega$ model, $\tilde{k}$ is set simply to $k$, which in the free stream is set to a small value to prevent a division by zero. We prefer, however, to specify $k = 0$ in the free stream and, in order to prevent a division by zero in the dissipation rate source term, to define $\tilde{k}$ as

$$\tilde{k} = \max\left[k,\ \min\left(k_{\text{div}},\ \frac{\omega\mu}{\rho}\right)\right], \tag{10}$$

where $k_{\text{div}}$ is a user-specified constant which is generally set lower than one-tenth of the maximum value of $k$ throughout the boundary layer. This is verified numerically not to affect the laminar sublayer but to improve the robustness and efficiency of the integration significantly. The minimum between $k_{\text{div}}$ and $\omega\mu/\rho$ is taken so that a clipping occurs *only* in nonturbulent flow regions in which an accurate representation of $\omega$ does not affect the accuracy of the flow field.

## 3. DISCRETIZATION

The discretization of all terms in the residual is now presented. The use of tensor form in writing the governing equations in curvilinear coordinates shown in Eq. (1), along with the unique compact $K_{ij}$ matrix described in Eq. (5), simplifies greatly the discretization and practical implementation of the viscous terms: fewer than 100 lines of code are needed to implement the viscous contribution of the residual in all dimensions. By referring to the discretized form of the derivatives along the $X_i$ coordinate as $\delta_{X_i}$, the discretized residual $R_\Delta$ can be written as

$$R_\Delta = \sum_{i=1}^{d}\left[\delta_{X_i} F_i - \sum_{j=1}^{d}\delta_{X_i}\left(K_{ij}\delta_{X_j}G\right)\right] - S, \tag{11}$$

where the diffusion terms are discretized using second-order-accurate centered finite-difference stencils which, should $i = j$, give

$$\left[\delta_{X_i}\left(K_{ii}\delta_{X_i}G\right)\right]^{X_i} = K_{ii}^{X_i+1/2}(G^{X_i+1} - G^{X_i}) - K_{ii}^{X_i-1/2}(G^{X_i} - G^{X_i-1}), \tag{12}$$

or, should $i \neq j$, give,

$$\begin{aligned}&\left[\delta_{X_i}\left(K_{ij}\delta_{X_j}G\right)\right]^{X_i,X_j}\\ &\quad= \frac{1}{4}K_{ij}^{X_i+1/2,X_j}(G^{X_i,X_j+1} + G^{X_i+1,X_j+1} - G^{X_i,X_j-1} - G^{X_i+1,X_j-1})\\ &\quad\quad - \frac{1}{4}K_{ij}^{X_i-1/2,X_j}(G^{X_i-1,X_j+1} + G^{X_i,X_j+1} - G^{X_i-1,X_j-1} - G^{X_i,X_j-1}),\end{aligned} \tag{13}$$

where $K^{X_i+1/2}$ midway between nodes is taken as half the value of $K^{X_i+1}$ and $K^{X_i}$, for example. The convection terms are discretized using a conservative form of the Roe scheme [18], turned second-order accurate through a symmetric minmod limiter by Yee *et al.* [19].

$$\left[\delta_{X_i}F_i\right]^{X_i} = \frac{1}{2}\left[F_i^{X_i+1} - F_i^{X_i-1} - \left[\frac{L_i^{-1}|\lambda_i|N_i}{J}\right]^{X_i+\frac{1}{2}} + \left[\frac{L_i^{-1}|\lambda_i|N_i}{J}\right]^{X_i-\frac{1}{2}}\right], \tag{14}$$

where $|\lambda_i|$ stands for the absolute value of the eigenvalues of the convective flux Jacobian $A_i \equiv \partial F_i/\partial Q$, and where

$$N_i^{X_i} = M_i^{X_i} - \operatorname{minmod}\left(M_i^{X_i-1}, M_i^{X_i}, M_i^{X_i+1}\right), \tag{15}$$

$$\text{with}\quad M_i^{X_i} = L_i^{X_i}((JQ)^{X_i+1/2} - (JQ)^{X_i-1/2}). \tag{16}$$

In the above, the properties at the interface are determined from Roe averaging [18]. For the minmod limiter to be in pseudo-control-volume form, it is necessary to ensure that all metric terms needed to construct the $N$ matrix at one cell interface are measured at that particular interface.

A small positive value can be added to the eigenvalues to fix the aphysical carbuncle phenomenon originating from the Roe scheme when tackling certain problems, especially blunt bodies. This is usually referred to as “entropy correction,” but it is just a convenient way of adding artificial dissipation to the flux discretization [20] and can significantly deteriorate the accuracy of the scheme in turbulent boundary layers (see, for example, results obtained by Parent and Sislian [21]). Unless otherwise specified, no entropy correction term is used in this paper.

Finally, all partial derivatives of the source terms are discretized using second-order-accurate three-point stencils, except for the stencil of the time derivative term, which is limited and is set to

$$\delta_t Q = \frac{1}{\Delta t}\left[Q^t - Q^{t-\Delta t} + \frac{1}{2}\,\text{minmod}(Q^t - Q^{t-\Delta t}, Q^{t-\Delta t} - Q^{t-2\Delta t}) - \frac{1}{2}\,\text{minmod}(Q^t - Q^{t-\Delta t}, Q^{t-\Delta t} - Q^{t-2\Delta t}, Q^{t-2\Delta t} - Q^{t-3\Delta t})\right], \tag{17}$$

where the minmod function returns the minimum of its arguments if the arguments are all positive, the maximum if the arguments are all negative, and zero if the arguments are of mixed signs. It is noted that the second-order contribution in Eq. (17) is in nonconservative form, but, to the authors' knowledge, such is unavoidable if no future value of $Q$ (at a time $t + \Delta t$) is included in the stencil and a second-order-accurate stencil that results in no spurious oscillations is desired. The nonconservation of the stencil is weak and numerical tests indicate that Eq. (17) performs well.

### 3.1. *Eigenstructure of the Convective Flux Jacobian*

Since the Roe scheme is used to discretize the convection derivatives, the determination of the eigenstructure of the convective flux Jacobian $A_i \equiv \partial F_i/\partial Q$ is needed. It can be checked by substitution that the following definition of $\lambda_i$,

$$\lambda_i = [V_i, V_i, \rightarrow, V_i + a\hat{X}_i, V_i - a\hat{X}_i, V_i, V_i]^{\mathrm{D}}, \tag{18}$$

satisfies the necessary relationship $\det(A_i - w_i I) = 0$ (with $w_i$ any element on the diagonal of $\lambda_i$) and is hence a valid eigenvalue matrix. Denoting the flow speed by $q$, the nonmetric effective speed of sound is found to be equal to

$$a = \left(P_\rho + \frac{2}{3}k + P_{\rho E}(H - q^2 - k)\right)^{\frac{1}{2}}, \tag{19}$$

which as might be expected is a function of the kinetic energy of turbulence, contrarily to the “nonturbulent” speed of sound (here denoted $a_{k=0}$) encountered in the eigenstructure of the convection terms of the molecular Navier–Stokes equations. Dividing both sides of Eq. (19) by $a_{k=0}$, and after some reformatting, the normalized speed of sound can be shown to be equal to

$$\bar{a} \equiv \frac{a}{a_{k=0}} = \left(1 + \frac{P_{\rho E} + 1}{3}\mathrm{M}_{\mathrm{t}}^2\right)^{\frac{1}{2}}, \tag{20}$$

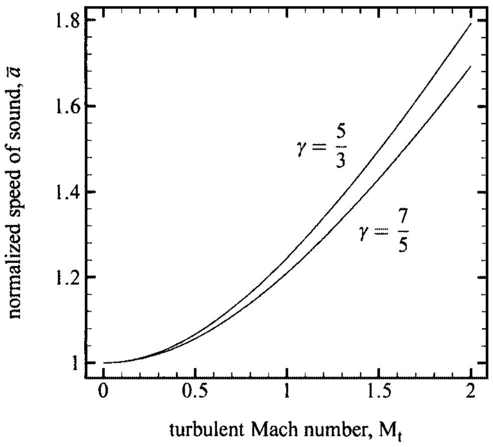

**FIG. 1.** Normalized speed of sound $\bar{a} = a/a_{k=0}$ versus the turbulent Mach number $M_t = \sqrt{2k}/a_{k=0}$, for a calorically perfect gas according to Eq. (20).

where for a perfect gas, $P_{\rho E} + 1$ is equal to the ratio of the specific heats, $\gamma$. Figure 1 shows the relationship between $\bar{a}$ and the turbulent Mach number for $\gamma = 7/5$ and $5/3$. As the turbulent Mach number increases, its influences on $\bar{a}$ becomes more predominant due to the relative speed of the turbulent vortices, with respect to the average vortex speed of displacement, gradually overtaking the thermodynamic sound speed as the information-propagation mechanism. For a perfect or real gas, the derivatives of pressure with respect to the mass-weighted total energy and density are equal to

$$P_{\rho E} = \frac{1}{\rho \frac{\partial}{\partial P} e(P, \rho)} \quad \text{and} \quad P_\rho = -P_{\rho E}\left(E - q^2 - k + \rho \frac{\partial}{\partial \rho} e(P, \rho)\right), \tag{21}$$

where $e$ is the internal energy of the gas and where it is assumed that any thermodynamic property can be obtained from only two others. The right eigenvectors are not unique, and each column of the matrix can be multiplied by a constant other than 0; here, we choose the multiplying constant of each column to keep the same units along each row, except for the last column which is further multiplied by $a^2/\omega$, which is found to result in faster convergence:

$$L_i^{-1} = \begin{bmatrix} 1 & 0 & \rightarrow & 1 & 1 & 0 & 0 \\ v_1 & l_i^{1,1} a & \rightarrow & v_1 + \frac{aX_{i,1}}{\hat{X}_i} & v_1 - \frac{aX_{i,1}}{\hat{X}_i} & 0 & 0 \\ \vdots & \vdots & \searrow & \vdots & \vdots & \vdots & \vdots \\ v_d & l_i^{d,1} a & \rightarrow & v_d + \frac{aX_{i,d}}{\hat{X}_i} & v_d - \frac{aX_{i,d}}{\hat{X}_i} & 0 & 0 \\ H - \frac{a^2}{P_{\rho E}} & \sum_j l_i^{j,1} a v_j & \rightarrow & H + \frac{aV_i}{\hat{X}_i} & H - \frac{aV_i}{\hat{X}_i} & a^2 - \frac{2a^2}{3P_{\rho E}} & 0 \\ k & 0 & \rightarrow & k & k & a^2 & 0 \\ \omega & 0 & \rightarrow & \omega & \omega & 0 & a^2 \end{bmatrix}. \tag{22}$$

The exactness of the right eigenvectors can be readily verified by the relation $\lambda_i = L_i A_i L_i^{-1}$. Note that the columns of the right eigenvectors containing $l_i^{m,n}$ are not needed in one dimension, while in two dimensions $l_i^{m,n}$ takes on the form

$$l_i^{m,1} = (-1)^{m+1} X_{i,m+1}/\hat{X}_i, \tag{23}$$

and in three dimensions it becomes

$$l_i^{m,1} = \frac{X_{i,m+2} - X_{i,m+1}}{\left[\sum_{j=1}^{3}(X_{i,j+1} - X_{i,j})^2\right]^{1/2}}, \quad l_i^{m,2} = \frac{X_{i,m+1} l_i^{m+1,1} - X_{i,m+2} l_i^{m+2,1}}{\hat{X}_i}. \tag{24}$$

In the above, $\hat{X}_i$ corresponds to the magnitude of all derivatives of $X_i$; that is,

$$\hat{X}_i = \left(\sum_{j=1}^{d} X_{i,j}^2\right)^{\frac{1}{2}}. \tag{25}$$

## 4. PSEUDO-TIME INTEGRATION

Using implicit Euler pseudo-time marching the delta form of the discretized equations can be shown to correspond to

$$\frac{\Delta^n Q}{\Delta\tau} + \sum_{i=1}^{d}\left[\delta_{X_i}\Delta^n F_i - \sum_{j=1}^{d}\delta_{X_i}(K_{i,j}\delta_{X_j}\Delta^n G)\right] - \Delta^n S = -R_\Delta, \tag{26}$$

where to minimize storage requirements and the inversion effort the LHS is approximated using a multiplication of one-dimensional operators based on a block-implicit approximate factorization algorithm [22, 23] and a linearization strategy of the viscous terms by Chang and Merkle [8],

$$\left[\prod_{i=1}^{d}\left(I + \Delta\tau\overline{\delta_{X_i} A_i} - \Delta\tau\sum_{j=1}^{d}\delta_{X_i}\left(K_{ij}\delta_{X_j}B\right) - \Delta\tau\delta_{1i}^{\mathrm{Kr}}C^-\right)\right]\Delta^n Q = -\Delta\tau R_\Delta, \tag{27}$$

where $\delta_{1i}^{\mathrm{Kr}}$ is the Kronecker delta, $B$ the linearization Jacobian of the viscous terms ($B \equiv \partial G/\partial Q$), and $C_i^-$ the linearization Jacobian of the negative source terms ($\partial S^-/\partial Q$) for the $i = 1$ sweep but ignored for the other sweeps. Only the negative source terms are linearized to ensure the stability of the implicit algorithm [24] and they are set to

$$S^- = \frac{1}{J}\begin{bmatrix} \vdots \\ -\rho k\omega \\ -\frac{5}{6}\rho k\omega^2/\tilde{k} \end{bmatrix} - \frac{\partial Q}{\partial t}. \tag{28}$$

The term $\overline{\delta_{X_i} A_i}$ is symbolic and stands for the linearization of the first-order Roe scheme with the Roe Jacobian locally frozen. The use of a fully linearized Roe scheme is shown in Batten *et al.* [20] not to decrease the number of iterations needed for convergence for several test problems (in some cases it is even detrimental) while requiring more work per iteration than the frozen Jacobian approach. Hence, the equation to solve at each node for the $i$th sweep can be written as

$$\left[-K_{ii}^{X_i-\frac{1}{2}}B^{X_i-1} - \frac{(J^{-1}L^{-1}|\lambda|L)_i^{X_i-\frac{1}{2}}}{2(J^{-1})^{X_i-1}} - \frac{A_i^{X_i-1}}{2}\right]\Delta\tilde{Q}_i^{X_i-1} + \left[\frac{I}{\Delta\tau^{X_i}} - \delta_{i1}(C^-)^{X_i}\right.$$
$$\left. + \left(K_{ii}^{X_i-\frac{1}{2}} + K_{ii}^{X_i+\frac{1}{2}}\right)B^{X_i} + \frac{(J^{-1}L^{-1}|\lambda|L)_i^{X_i-\frac{1}{2}} + (J^{-1}L^{-1}|\lambda|L)_i^{X_i+\frac{1}{2}}}{2(J^{-1})^{X_i}}\right]\Delta\tilde{Q}_i^{X_i}$$
$$+ \left[-K_{ii}^{X_i+\frac{1}{2}}B^{X_i+1} - \frac{(J^{-1}L^{-1}|\lambda|L)_i^{X_i+\frac{1}{2}}}{2(J^{-1})^{X_i+1}} + \frac{A_i^{X_i+1}}{2}\right]\Delta\tilde{Q}_i^{X_i+1} = \frac{I}{\Delta\tau^{X_i}}\Delta\tilde{Q}_{i-1}^{X_i}, \quad (29)$$

where $\Delta\tilde{Q}_0^{X_i} = -\Delta\tau^{X_i}R_\Delta^{X_i}$ and the total flux increment $\Delta Q^{X_i}$ is set to $\Delta\tilde{Q}_d^{X_i}$.

It is emphasized that the success of approximate factorization relies on the degree of invariance of the linearization matrices, deterring the inclusion of a linearized form of the minmod limiter on the implicit side. Numerical experiments show that a “switch” type of algorithm on the implicit side might induce erratic patterns in the convergence history, sometimes preventing a converged solution altogether. For similar reasons, the implicit treatment of the cross-diffusion terms is not recommended, as their linearization necessarily involves spatial derivatives which are subject to change from iteration to iteration.

Block-implicit approximate factorization is chosen here as it is still one of the most used techniques for solving the compressible Navier–Stokes equations in the hypersonic range. However, it is noted that the use of a different pseudo-time-marching algorithm (such as DDADI [12], MAF($k$) [25], or LUSGS [26]) has been observed by the authors not to affect the performance gains obtained with the marching-window algorithm presented herein.

## 4.1. *Local Pseudo-Time Step*

One commonly used acceleration technique is local pseudo-time stepping based on the CFL condition which results in a wave traveling speed of one node per iteration for convection-dominated flows. However, in multiple dimensions, each dimension assumes a different CFL condition and one faces the dilemma of specifying a wave traveling speed proportional to the dimension exhibiting the lowest CFL condition, commonly referred to as a minimum CFL-based local time step, or to the dimension exhibiting the highest CFL condition, which is referred to as a maximum CFL-based local time step. A formulation including both the minimum and maximum CFL-based approaches can take the form

$$\Delta\tau = \text{CFL}\ \max_{i=1}^{d}\left(\frac{1}{|V_i| + a\hat{X}_i}\right)^{\sigma}\ \min_{i=1}^{d}\left(\frac{1}{|V_i| + a\hat{X}_i}\right)^{1-\sigma}, \quad (30)$$

where a $\sigma$ varying between 0 and 1 induces a time step of a magnitude situated, respectively, between a minimum and a maximum CFL-based time step. While it is acknowledged that for viscous-dominated regions a local time step based on the Von Neumann number (VNN) would result in a more equitable wave propagation, which might translate into faster convergence, for the purposes of this paper Eq. (30) is used exclusively.

## 5. BOUNDARY CONDITIONS

A multiblock stratagem is generally required when tackling complex geometries with a structured mesh, but it can significantly complicate the implementation of the domain decomposition algorithms presented herein for reasons that shall become apparent shortly. As a substitute to using multiple blocks connected to the geometry and to one another through their outer edges (or planes in 3D), any node that is part of the computational domain is allowed to be either a boundary, inner, or inactive node. Although not as multipurpose as the multiblock, such an approach can be used to solve a wide variety of flow fields while retaining all the simplicity of a single block. Figure 2 shows, for example, how the node types would be distributed for a backward-facing step and a two-element airfoil.

Zeroth-order extrapolation polynomials are used to obtain the properties from the adjacent inner node at the supersonic outflow boundary (hereafter referred to simply as outflow boundary), while the properties at the supersonic inflow (hereafter referred to as inflow), are unaltered in pseudotime. At the symmetry boundary node, a first-order extrapolation polynomial of the form

$$\psi^X = \frac{4}{3}\psi^{X+1} - \frac{1}{3}\psi^{X+2} \tag{31}$$

is employed to obtain $P^\star$, $k$, $\omega$, $\rho$, and the velocity components tangent to the surface, while the perpendicular velocity component is set to zero. At the wall, the turbulence kinetic energy and the velocity are fixed to zero, while the effective pressure and temperature (in the case of an adiabatic wall) are extrapolated as in Eq. (31). Also, following Wilcox [27], the dissipation rate at the wall is specified to

$$\omega_{\mathrm{w}} = \frac{36}{5}\frac{\mu}{\rho d_{\mathrm{w}}^2}, \tag{32}$$

with $d_{\mathrm{w}}$ the distance between the wall node and its nearest neighbor.

It is well-known that an implicit treatment of the boundary nodes results in less prohibitive restrictions on the pseudo-time-step size for some problems, but when solving strong shock waves or other highly nonlinear phenomena it is not uncommon for the time-step size to be limited in any case by the flow physics, even if the time-stepping scheme can be shown

```
1111111111111111111111111111111111111111111      11111111111111111111111111111111111111111111111
0+++++++++++++++++++++++++++++++++++++++++1      0+++++++++++++++++++++++++++++++++++++++++++++1
0+++++++++++++++++++++++++++++++++++++++++1      0+++++++++++++++++++++++++++++++++++++++++++++1
0+++++++++++++++++++++++++++++++++++++++++1      0+++++++++++++++++++++++++++++++++++++++++++++1
0+++++++++++++++++++++++++++++++++++++++++1      0+++++++++++++++++++++++++++++++++++++++++++++1
0+++++++++++++++++++++++++++++++++++++++++1      0++++++33333333333333+++++++++++++++++++++++++1
0+++++++++++++++++++++++++++++++++++++++++1      0++++++3............3+++++++++++++++++++++++++1
0+++++++++++++++++++++++++++++++++++++++++1      0++++++3............3+++++++++++++++++++++++++1
0+++++++++++++++++++++++++++++++++++++++++1      0++++++3............3+++++++++++++++++++++++++1
0+++++++++++++++++++++++++++++++++++++++++1      0++++++3............3++++33333333333++++++++++1
33333333333333333333++++++++++++++++++++++1      0++++++3............3++++3.........3++++++++++1
...................3++++++++++++++++++++++1      0++++++3............3++++3.........3++++++++++1
...................3++++++++++++++++++++++1      0++++++33333333333333++++3.........3++++++++++1
...................3++++++++++++++++++++++1      0++++++++++++++++++++++++3.........3++++++++++1
...................3++++++++++++++++++++++1      0++++++++++++++++++++++++3.........3++++++++++1
...................3++++++++++++++++++++++1      0++++++++++++++++++++++++3.........3++++++++++1
...................3++++++++++++++++++++++1      0++++++++++++++++++++++++33333333333++++++++++1
...................3++++++++++++++++++++++1      0+++++++++++++++++++++++++++++++++++++++++++++1
...................3++++++++++++++++++++++1      0+++++++++++++++++++++++++++++++++++++++++++++1
...................3++++++++++++++++++++++1      0+++++++++++++++++++++++++++++++++++++++++++++1
...................33333333333333333333333      11111111111111111111111111111111111111111111111
```

**FIG. 2.** Examples of the distribution of the node types for a backward-facing step (left) and a two-element airfoil (right). The number 3 represents a wall condition; 0, inflow; 1, outflow; +, inner nodes; and dots, unactivated nodes.

to be Von Neumann unconditionally stable (see the chapter on nonlinear stability in Laney [28]). Moreover, experience shows that treating the boundary conditions presented herein in an explicit manner does not restrict the size of the local time step more than implicit boundary conditions would. For all numerical experiments presented, an explicit treatment of the boundary nodes is chosen.

## 6. DOMAIN DECOMPOSITION ALGORITHMS

While domain decomposition is generally used for parallel computing purposes or used to enable the implementation of different discretization/integration methods in different subdomains, it is utilized here as a means of accelerating the convergence of quasihyperbolic systems. We refer to an elliptic system of equations as being quasihyperbolic when some of the terms, but not all, can be regrouped to form a hyperbolic set of equations, and whose solution is very close to the solution of the hyperbolic set of terms. For instance, the steady-state Navier–Stokes equations in the hypersonic regime away from the surfaces would exhibit a weak influence of the diffusion terms (responsible for the ellipticity of the system) on the solution compared to the convection terms (the hyperbolic set) and would hence be classified as quasihyperbolic. Similarly, we refer to an elliptic system of equations as being quasiparabolic when some of the terms, but not all, can be regrouped to form a set of parabolic equations, and whose solution is very close to the solution of the parabolic set of terms. The Favre-averaged Navier–Stokes equations closed by the $k\omega$ model solved at steady state over a turbulent flat plate would be termed quasiparabolic, as the streamwise diffusion terms and the upstream component of the convection terms play a negligible role compared to the other terms.

The acceleration techniques presented in this paper are aimed at reducing the work needed to solve quasihyperbolic or quasiparabolic systems through the use of domain decomposition. Nonetheless, the effectiveness of the methods is not limited to entirely quasihyperbolic/parabolic systems and extends to systems where some regions are quasihyperbolic/parabolic and others strictly elliptic. It is emphasized that domain decomposition is used here solely as a convergence acceleration technique and *does not* modify the discretized residual, the time-stepping schemes, and the convergence criterion, except in the case of the active-domain method. Convergence is attained independently of the acceleration technique when

$$\xi \leq \xi_{\text{verge}} \quad \forall \text{ inner nodes,} \tag{33}$$

where $\xi$ is a convergence criterion based on the maximum between the discretized continuity and energy residuals,

$$\xi \equiv \max\left(\frac{\left|R_\Delta^{\text{continuity}}\right|}{J^{-1}\rho}, \frac{\left|R_\Delta^{\text{energy}}\right|}{J^{-1}\rho E}\right), \tag{34}$$

which is divided by $Q$ to obtain units involving only pseudotime (i.e., $\frac{1}{\text{s}}$). The user-defined convergence threshold $\xi_{\text{verge}}$ is typically given a value of $100\ \frac{1}{\text{s}}$, yet this value is not universal and is dependent on the flow field at hand. Based on dimensional analysis arguments, $\xi_{\text{verge}}$ can be thought of as the inverse of a time scale common for all nodes, which we formulate

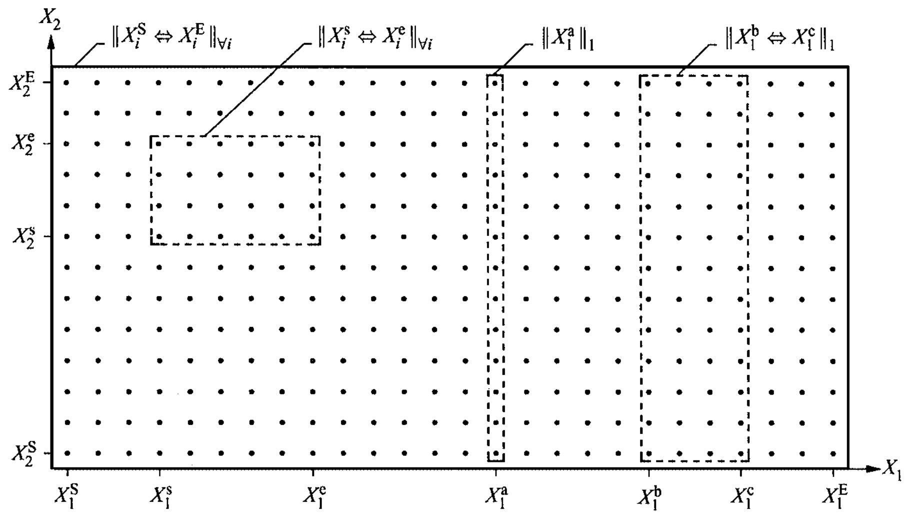
$X_2$
$\|X_i^S \Leftrightarrow X_i^E\|_{\forall i}$
$\|X_i^s \Leftrightarrow X_i^e\|_{\forall i}$
$\|X_1^a\|_1$
$\|X_1^b \Leftrightarrow X_1^c\|_1$
$X_2^E$
$X_2^e$
$X_2^s$
$X_2^S$
$X_1$
$X_1^S$
$X_1^s$
$X_1^e$
$X_1^a$
$X_1^b$
$X_1^c$
$X_1^E$

**FIG. 3.** Example of computational domain and subdomain notation in two dimensions. The computational domain limits are denoted by the superscripts E and S.

in terms of the free-stream flow speed and a characteristic length, i.e.,

$$\xi_{\text{verge}} \sim \frac{1}{10}\frac{q_\infty}{L_c}, \tag{35}$$

where an analogy can be made to the time needed to obtain steady-state flow using an experimental setup. The characteristic length $L_c$ can be taken as the length of the domain, for instance. It is noted, however, that the efficiency of the domain decomposition methods presented herein is dependent on the precision of $\xi$ as a convergence criterion, and since this varies from one flow problem to the next, it might not always be possible to achieve at first the proper compromise between optimal convergence rate and acceptable accuracy by using Eq. (35).

Identifying the limits of the computational domain by $X_i^S$ and $X_i^E$, with $i \in [1, \ldots, d]$, and the limits of a subdomain by $X_i^s$ and $X_i^e$, with $i \in [1, \ldots, d]$, the region spanned by the subdomain is referred to by the notation $\|X_i^s \Leftrightarrow X_i^e\|_{\forall i}$, as shown in Fig. 3. For a subdomain with limits different from the computational domain limits in only one dimension the notation $\|X_n^s \Leftrightarrow X_n^e\|_n$ is employed, where it is implied that the limits in the dimensions other than the $n$th do not differ from those of the computational domain (see Fig. 3). Also, $\|X_n^s\|_n$ is a shortcut that stands for the subdomain $\|X_n^s \Leftrightarrow X_n^s\|_n$. A property that is used in conjunction with the domain decomposition algorithms is the number of nodes of dependence of the discretized residual, $b_r$, which is defined as

$$b_r \equiv \begin{cases} \text{the maximum number of nodes on which the discretized} \\ \quad \text{residual depends on each side of the center node,} \\ \text{or} \\ \text{half the maximum discretization stencil point minus one} \\ \quad \text{if the stencil is symmetric.} \end{cases} \tag{36}$$

For example, the minmod TVD discretization stencil (which is the longest of all stencils contained in the residual) would give $b_r = 2$, but should a first-order Roe scheme be employed instead, then $b_r$ would be set to one. Similarly, the number of nodes of dependence of the boundary nodes is defined as

$$b_b \equiv \left\{ \begin{array}{l} \text{the maximum number of nodes any boundary node depends} \\ \quad \text{on along one direction,} \end{array} \right. \tag{37}$$

which is set to 2, since the properties at the boundary nodes are extrapolated from at most two inner nodes using a blend of zeroth- and first-order extrapolation polynomials.

When the nodes comprised in the subdomain $\|X_i^{\rm s} \Leftrightarrow X_i^{\rm e}\|_{\forall i}$ are updated in pseudotime, then it follows from the definition of $b_b$ that the boundary nodes situated inside $\|X_i^{\rm s} - b_b \Leftrightarrow X_i^{\rm e} + b_b\|_{\forall i}$ must be updated. The residual, which depends on both inner and boundary nodes, must then be updated between $\|X_i^{\rm s} - b_b - b_r \Leftrightarrow X_i^{\rm e} + b_b + b_r\|_{\forall i}$. In many cases where there are no boundary nodes situated in the region $\|X_i^{\rm s} - b_b \Leftrightarrow X_i^{\rm e} + b_b\|_{\forall i}$, it is sufficient to update the residual in $\|X_i^{\rm s} - b_r \Leftrightarrow X_i^{\rm e} + b_r\|_{\forall i}$. For all methods presented in this paper, however, this shortcut is not implemented.

## 6.1. *Standard Cycle*

The “standard cycle” here implies the usual way of updating the solution in pseudotime, by first finding the residual for all nodes and then updating the solution. The algorithm can be written in the following steps:

1. Update the boundary nodes in the domain $\|X_i^{\rm S} \Leftrightarrow X_i^{\rm E}\|_{\forall i}$.
2. Update the residual in the domain $\|X_i^{\rm S} \Leftrightarrow X_i^{\rm E}\|_{\forall i}$.
3. Update $Q$ (by pseudo-time stepping) in the domain $\|X_i^{\rm S} \Leftrightarrow X_i^{\rm E}\|_{\forall i}$.
4. Attain convergence when $\xi \le \xi_{\rm verge}$ in the domain $\|X_1^{\rm S} \Leftrightarrow X_1^{\rm E}\|_{\forall i}$.

## 6.2. *Multizone Cycle*

One strategy toward improving the standard cycle is to divide the computational domain into a number of nonoverlapping zones of approximately equal size and to update in pseudotime only the zones in which $\xi > \xi_{\rm verge}$. This stratagem has been previously employed by Sawley and Tegner [29] as a convergence acceleration technique for supersonic flows, but where the computational domain is split into several blocks, instead of several zones. Note that a “zone” is defined as a computational domain region that can be bounded by boundary and/or inner nodes (see, for example, Ref. [30]), while a “block” is defined as a region delimitated by boundary nodes only. The zone length in each dimension is set to at most $\phi_1$, a user-specified constant usually given a value of 20. At each iteration, should the maximum $\xi$ inside each zone be greater than the user-specified threshold $\xi_{\rm verge}$, the inner nodes up to the zone boundaries are updated in pseudotime, followed by the update of the boundary nodes up to the zone boundaries expanded by $b_b$, and the update of the residual up to the zone boundaries expanded by $b_b + b_r$. The residual and properties of all other nodes of the computational domain are not altered. Prior to the first iteration, the computational domain is divided into a number of nonoverlapping zones of a length in each dimension no greater than $\phi_1$, with each zone $z$ defined by the subdomain $\|X_i^{z,\rm s} \Leftrightarrow X_i^{z,\rm e}\|_{\forall i}$. Then, at each iteration, the following steps are performed:

1. For each zone $z$, update $Q$ (by pseudo-time stepping) in the subdomain $\|X_i^{z,\rm s} \Leftrightarrow X_i^{z,\rm e}\|_{\forall i}$ if $\xi > \xi_{\rm verge}$ in the subdomain $\|X_i^{z,\rm s} \Leftrightarrow X_i^{z,\rm e}\|_{\forall i}$.

2. For each zone $z$, update the boundary nodes in the subdomain $\|X_i^{z,\mathrm{s}} - b_b \Leftrightarrow X_i^{z,\mathrm{e}} + b_b\|_{\forall i}$ if $\xi > \xi_{\mathrm{verge}}$ in the subdomain $\|X_i^{z,\mathrm{s}} \Leftrightarrow X_i^{z,\mathrm{e}}\|_{\forall i}$.
3. For each zone $z$, update the residual in the subdomain $\|X_i^{z,\mathrm{s}} - b_b - b_r \Leftrightarrow X_i^{z,\mathrm{e}} + b_b + b_r\|_{\forall i}$ if $\xi > \xi_{\mathrm{verge}}$ in the subdomain $\|X_i^{z,\mathrm{s}} \Leftrightarrow X_i^{z,\mathrm{e}}\|_{\forall i}$.
4. Attain convergence when $\xi \leq \xi_{\mathrm{verge}}$ in the domain $\|X_1^{\mathrm{S}} \Leftrightarrow X_1^{\mathrm{E}}\|_{\forall i}$.

It is noted that the multizone cycle ensures the residual on all nodes are up-to-date after each iteration but, due to the non-self-starting property of this cycle, it is necessary to compute the residual on the entire domain before the first iteration is performed.

## 6.3. *Active-Domain Cycle*

The active domain is an algorithm aimed at decreasing the work needed for convergence of supersonic inviscid flow [13] and refers to a bandlike computational domain marching in the flow direction in which localized pseudo-time stepping is performed. The domain width automatically adjusts to the size of subsonic regions when encountered by monitoring the streamwise component of the Mach number, as shown in Fig. 4. In our notation, the active-domain algorithm can be written as follows, denoting the left boundary of the computational window by $X_1^{\mathrm{s}}$ and the right boundary by $X_1^{\mathrm{e}}$:

1. Update the boundary nodes in the subdomain $\|X_1^{\mathrm{s}} \Leftrightarrow X_1^{\mathrm{e}}\|_1$.
2. Update the residual in the subdomain $\|X_1^{\mathrm{s}} \Leftrightarrow X_1^{\mathrm{e}}\|_1$.
3. Update $Q$ (by pseudo-time stepping) in the subdomain $\|X_1^{\mathrm{s}} \Leftrightarrow X_1^{\mathrm{e}}\|_1$.
4. Redefine the active-domain boundaries:

(a) if $M_1 < 1.001$ for any node in the subdomain $\|X_1^{\mathrm{s}} \Leftrightarrow X_1^{\mathrm{s}}\|_1$ then decrease $X_1^{\mathrm{s}}$ by one;

(b) if $M_1 < 1.001$ for any node in the subdomain $\|X_1^{\mathrm{e}} - (\phi_3 - \phi_0) \Leftrightarrow X_1^{\mathrm{e}}\|_1$ then increment $X_1^{\mathrm{e}}$ by one;

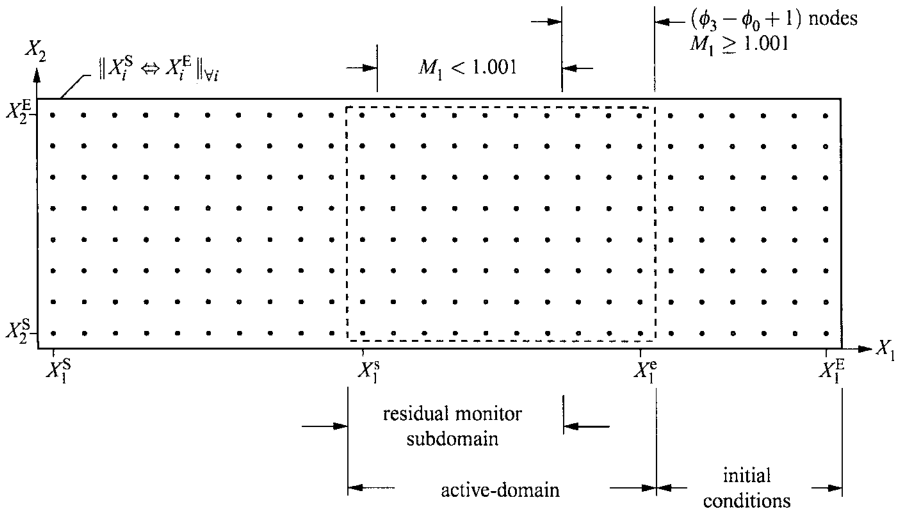

FIG. 4. Schematic of the active-domain cycle with the upstream and downstream boundaries in equilibrium surrounding an embedded subsonic region. $\phi_0$ is the minimum width of the residual monitor subdomain and $\phi_3$ is the minimum width of the active domain (when no subsonic region is present).

(c) if $\xi \leq \xi_{\text{verge}}$ for all nodes in the subdomain $\|X_1^{\text{s}} \Leftrightarrow X_1^{\text{e}} - (\phi_3 - \phi_0)\|_1$ then increment $X_1^{\text{e}}$ by $\phi_0$ and set $X_1^{\text{s}} = X_1^{\text{e}} - \phi_3$.

5. Attain convergence when $\xi \leq \xi_{\text{verge}}$ for all nodes in the subdomain $\|X_1^{\text{s}} \Leftrightarrow X_1^{\text{e}}\|_1$ and when $X_1^{\text{e}} = X_1^{\text{E}}$.

The size of the residual-monitor region $\phi_0$ and the size of the active-domain $\phi_3$ are user-specified constants typically given values of 4 and 9, respectively. It is emphasized that the active domain is restricted to inviscid flow due to the "ellipticity sensors" in Steps 4a and 4b, being based on the streamwise component of the Mach number. For viscous flows, this would effectively enlarge the active domain to the size of any object due to the vanishing value of the Mach number in the vicinity of a wall. Aside from being restricted to inviscid flow, the active-domain algorithm does not guarantee that $\xi \leq \xi_{\text{verge}}$ for all nodes of the computational domain when convergence is attained. This is due to the assumption in Step 4a that streamwise ellipticity is present locally only when the streamwise component of the Mach number is less than 1. For inviscid flow, this is exactly true if the discretization stencil of the streamwise convection derivative is upwinded (such as the first-order-accurate Roe scheme) but is not true for the Yee–Roe scheme used herein due to the Yee flux limiter being a function of downstream nodes, even when the flow is locally supersonic. Therefore, when used in conjunction with a flux limiter inducing streamwise ellipticity in supersonic flow, the active domain does not meet the necessary convergence criterion for a well-posed acceleration technique [as stated previously in Eq. (33)].

## 6.4. *Marching-Window Cycle*

An alternate form of the active-domain cycle that permits the solution of viscous streamwise separated flows and satisfies the convergence criterion of Eq. (33) is here presented. Named the marching window, the algorithm differs from the active domain on three points: (i) a dynamic outflow boundary is forced at the downstream boundary of the marching window (see Fig. 5), (ii) the ellipticity sensor responsible for a shift downstream of the

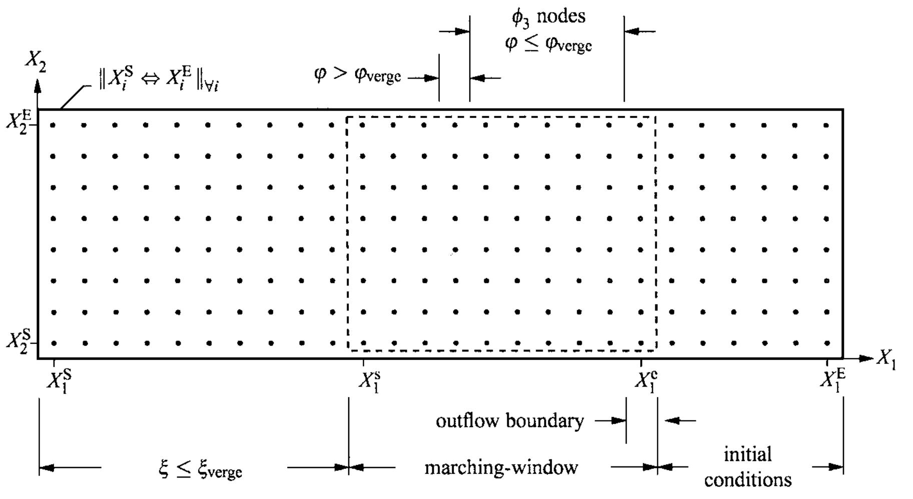

**FIG. 5.** Schematic of the marching-window cycle with the upstream and downstream boundaries in equilibrium surrounding an embedded streamwise elliptic region which is bounded upstream by the condition $\xi \leq \xi_{\text{verge}}$ and downstream by the condition $\varphi \leq \varphi_{\text{verge}}$. A dynamic outflow boundary condition is forced on all inner nodes in $\|X_1^{\text{e}}\|$.

downstream boundary of the marching window is based on a Vigneron splitting of the streamwise pressure derivative instead of the streamwise component of the Mach number, and (iii) the upstream boundary of the marching window is positioned such that $\xi \leq \xi_{\text{verge}}$ for all nodes upstream, instead of being a function of a residual monitor region and a streamwise ellipticity sensor based on the streamwise component of the Mach number. At the first iteration, the upstream boundary of the marching window is set to the upstream boundary of the computational domain, with the downstream boundary of the marching window separated from the upstream boundary by $b_b$ nodes. Denoting the upstream boundary of the marching window by $X_1^{\text{s}}$ and the downstream boundary by $X_1^{\text{e}}$, the marching-window cycle can be written as follows:

1. Update $Q$ (by pseudo-time stepping) in the subdomain $\|X_1^{\text{s}} \Leftrightarrow X_1^{\text{e}}\|_1$.
2. Update the boundary nodes in the subdomain $\|X_1^{\text{s}} - b_b \Leftrightarrow X_1^{\text{e}}\|_1$.
3. Update the residual (hence, $\xi$) in the subdomain $\|X_1^{\text{s}} - b_b - b_r \Leftrightarrow X_1^{\text{e}}\|_1$.
4. Redefine the marching-window boundaries:
   (a) find the maximum value for $X_1^{\text{s}}$ such that $\xi \leq \xi_{\text{verge}}$ for all nodes in the subdomain $\|X_1^{\text{S}} \Leftrightarrow X_1^{\text{s}} - 1\|_1$;
   (b) every $\phi_2$ iterations, if $\varphi > \varphi_{\text{verge}}$ for any node in the subdomain $\|X_1^{\text{e}} - \phi_3 \Leftrightarrow X_1^{\text{e}}\|_1$ or if $X_1^{\text{s}} > X_1^{\text{e}} - \phi_3$, then (i) increment $X_1^{\text{e}}$ by one, (ii) update the boundary nodes in the subdomain $\|X_1^{\text{e}} - 1 - b_b \Leftrightarrow X_1^{\text{e}}\|_1$, and (iii) update the residual in the subdomain $\|X_1^{\text{e}} - 1 - b_b - b_r \Leftrightarrow X_1^{\text{e}} - 1\|_1$.
5. Attain convergence when $\xi \leq \xi_{\text{verge}}$ for all nodes in the subdomain $\|X_1^{\text{s}} \Leftrightarrow X_1^{\text{e}}\|_1$ and when $X_1^{\text{e}} = X_1^{\text{E}}$.

The marching-window cycle is not self-starting and it must be ensured that the residual is updated for all nodes part of the computational window before the first iteration.

The ability of the marching window to satisfy the convergence criterion of Eq. (33) lies in Steps 1–3, where $Q$ is updated in pseudo-time, in Step 1, *before* determining the residual, in Step 3. Once $Q$ is updated in Step 1 on the subdomain $\|X_1^{\text{s}} \Leftrightarrow X_1^{\text{e}}\|_1$, since a boundary node depends on at most $b_b$ neighbors, it is sufficient to update the boundary nodes in the subdomain $\|X_1^{\text{s}} - b_b \Leftrightarrow X_1^{\text{e}}\|_1$ to guarantee that all boundary nodes upstream of $X_1^{\text{e}}$ are up-to-date after Step 2. Once the boundary nodes have been updated in the subdomain $\|X_1^{\text{s}} - b_b \Leftrightarrow X_1^{\text{e}}\|_1$, since the discretized residual depends on at most $b_r$ neighbors, it is sufficient to update the residual in the subdomain $\|X_1^{\text{s}} - b_b - b_r \Leftrightarrow X_1^{\text{e}}\|_1$ to guarantee that the residual of all nodes upstream of $X_1^{\text{e}}$ are up-to-date after Step 3. Since $\xi$ is a function of the residual, $\xi$ upstream of $X_1^{\text{e}}$ is up-to-date, and the upstream boundary of the marching-window $X_1^{\text{s}}$ can be positioned correctly in Step 4a by ensuring for all nodes upstream of $X_1^{\text{e}}$ that $\xi \leq \xi_{\text{verge}}$. This serves two purposes: (i) the convergence criterion of Eq. (33) is satisfied if convergence is attained in Step 5, and (ii) the upstream boundary of the marching window moves upstream for any upstream propagating wave that affects the residual significantly and raises $\xi$ above the user-defined convergence threshold $\xi_{\text{verge}}$. Contrarily to the active domain, the upstream propagating wave is not limited to locally subsonic flow but includes all *significant* streamwise elliptic phenomena, such as streamwise separated flow, streamwise viscous derivatives, or flux limiters in the streamwise convection derivative, for instance.

Step 4b advances the marching-window downstream boundary when the width of the window is smaller than a user-specified constant $\phi_3$, or when the streamwise ellipticity sensor $\varphi$ is greater than the user-specified constant $\varphi_{\text{verge}}$ for any node part of the subdomain $\|X_1^{\text{e}} - \phi_3 \Leftrightarrow X_1^{\text{e}}\|_1$. The streamwise ellipticity sensor $\varphi$ is here chosen as the component of the streamwise convection derivative inducing a streamwise ellipticity. This is derived, following an approach by Vigneron *et al.* [1], by multiplying by $\zeta$ the effective-pressure terms in the momentum-fluxes part of the streamwise convection flux $F_1$. The eigenvalues of the streamwise convective flux Jacobian with $\zeta$ frozen can then be shown to correspond to

$$\tilde{\lambda}_1\left[V_1, V_1, \rightarrow, V_1+\frac{1}{2}V_1P_{\rho E}(1-\zeta)+\tilde{a}_1\hat{X}_1, V_1+\frac{1}{2}V_1P_{\rho E}(1-\zeta)-\tilde{a}_1\hat{X}_1, V_1, V_1\right]^{\mathrm{D}},$$

$$\text{with}\quad \tilde{a}_1=\left\{\frac{1}{4}[V_1P_{\rho E}(1-\zeta)]^2+\hat{X}_1^2\zeta\left[P_\rho+\frac{2}{3}k+P_{\rho E}(H-k-q^2)\right]\right\}^{\frac{1}{2}}.$$

Then, for all the eigenvalues to share the same sign (a necessary condition for a hyperbolic system), it is required that

$$\zeta=\min\left(1, V_1^2\frac{1+P_{\rho E}}{\hat{X}_1^2a^2+V_1^2P_{\rho E}}\right)=\min\left(1, \frac{M_1^2(1+P_{\rho E})}{1+M_1^2P_{\rho E}}\right), \tag{38}$$

where the streamwise Mach number $M_1$ corresponds to $V_1/a\hat{X}_1$. If multiplying by $\zeta$ the pressure-derivative-terms part of the momentum components of $\partial F_1/\partial X_1$ results in a hyperbolic system, it follows that the component of the streamwise convection derivative which induces a streamwise ellipticity is $(1-\zeta)$ times the pressure-derivative-terms part of the momentum components of $\partial F_1/\partial X_1$. The product is then normalized with $\rho a$ to obtain units of inverse pseudotime:

$$\varphi\equiv\frac{1}{\rho a}\left\{\sum_{j=1}^{d}\left[(1-\zeta)X_{1,j}\frac{\partial P^\star}{\partial X_1}\right]^2\right\}^{\frac{1}{2}}=\frac{\hat{X}_1}{\rho a}\max\left(0, \frac{1-M_1^2}{1+P_{\rho E}M_1^2}\right)\left|\frac{\partial P^\star}{\partial X_1}\right|. \tag{39}$$

The ellipticity sensor $\varphi$ makes two important assumptions: (i) the streamwise ellipticity originating from the streamwise viscous derivative terms and the flux-limiter part of the streamwise convection derivative is assumed negligible; and (ii) at the point where $\varphi$ is evaluated, the solution is assumed to be converged to steady state. The first assumption is remedied by introducing a minimum width of the marching window, $\phi_3$, which is typically given a value ranging from 9 to 15. The second assumption can lead to some performance degradation of the marching window when the flow near the downstream boundary is far from convergence. For this reason, the user-adjustable parameter $\phi_2$ is introduced in Step 4a, with the consequence of evaluating $\varphi$ every $\phi_2$ iterations only. Therefore, a high value given to $\phi_2$ helps to ensure a more converged solution near the downstream boundary and reduces the error in the ellipticity sensor $\varphi$ due to temporarily non-steady-state flow. It is suggested the ellipticity sensor threshold, $\varphi_{\text{verge}}$, be given a value of about 100 times the one given to $\xi_{\text{verge}}$; that is,

$$\varphi_{\text{verge}}\sim 10\frac{q_\infty}{L_{\mathrm{c}}}, \tag{40}$$

with $L_{\mathrm{c}}$ a characteristic length of the system. In Step 4b, after the downstream boundary of the marching window is advanced by one station, the update of the boundary nodes in the subdomain $\|X_1^{\mathrm{e}}-1-b_b \Leftrightarrow X_1^{\mathrm{e}}\|_1$ and of the residual in the subdomain $\|X_1^{\mathrm{e}}-1-b_b-b_r \Leftrightarrow X_1^{\mathrm{e}}-1\|_1$ is necessary to ensure that the residual is properly updated in the marching window, which is necessary for Step 1 to be performed correctly at the following iteration.

While the user-definable constants $\phi_2$, $\phi_3$, and $\varphi_{\text{verge}}$ affect the performance of the marching window cycle as a convergence acceleration technique, they do *not* affect the accuracy of the solution when convergence is attained due to the convergence criterion of Eq. (33) being satisfied.

## 6.5. *Marching Window/Multizone Cycle*

The performance of the marching-window algorithm can be enhanced by introducing multizone decomposition inside the marching window. Before each iteration, the marching-window subdomain $\|X_1^{\text{s}} \Leftrightarrow X_1^{\text{e}}\|_1$ is decomposed into several zones of length no more than $\phi_1$ nodes in each dimension. Then, Steps 1–3 of the marching-window cycle (see Section 6.4.) are replaced by Steps 1–3 of the multizone cycle (see Section 6.2).

## 6.6. *Sweeping Window/Multizone Cycle*

Intended for time-accurate simulations using dual-time stepping, the sweeping-window algorithm is identical to the marching-window algorithm, with the exception that no outflow boundary condition is forced on the downstream boundary of the sweeping window. When the converged solution of the previous time level is used as initial conditions for the current time level, not forcing an outflow condition at the downstream boundary helps to attain faster convergence due to the initial conditions providing a better "guess" at the downstream boundary. For the same reasons, the sweeping-window cycle can also be used to gain extra orders of magnitude of convergence on the solution obtained by the marching window.

## 7. NUMERICAL EXPERIMENTS

Three steady-state supersonic flow fields and one unsteady flow field are solved using the different types of cycles mentioned in the last section, and the performance of each is assessed on the basis of (i) the number of effective iterations, (ii) CPU time, and (iii) maximum storage required. To enable a fair comparison between the different cycle strategies, the number of effective iterations is defined as

$$\text{effective iterations} \equiv \frac{\text{number of times an inner node is updated}}{\text{total number of inner nodes}}, \tag{41}$$

which is a good measure of the cycle performance as long as most of the computing effort is spent on the pseudo-time stepping instead of the residual, due to the overlap of the residual determination when a multizone decomposition is used. The implicit scheme used herein spends three-quarters of its computing effort on the time-stepping side, thereby reducing the residual overlap overhead work and justifying the use of Eq. (41) as a performance parameter. In spite of being accurately measured, the number of CPU seconds is not regarded as a more meaningful performance parameter due to the unavoidable bias that might occur in the programming of the cycles and the high dependence of the work on the architecture of the computer. Certain enhancements to the multizone cycle, such as unifying adjacent zones, could be implemented which would result in a nonnegligible decrease in work, while the use of a vector computer (of CRAY type) would be an advantage to the longer loops present in the standard cycle. Therefore, both the number of effective iterations and the CPU time are monitored for all test cases.

### 7.1. *Inviscid Supersonic Inlet with a Blunt Leading Edge*

A first comparison between the different cycles is performed for a steady-state inviscid flow over a 1-m-long supersonic inlet. Air enters the channel at a Mach number of 5, a pressure of 4 kPa, and a temperature of 240 K. The grid size is varied between $128 \times 64$ and $512 \times 256$ nodes. The user-defined parameters of interest are set to (when applicable)

$$\sigma = 0.5, \quad \xi_{\text{verge}} = 100\ \frac{1}{\text{s}}, \quad \varphi_{\text{verge}} = 5000\ \frac{1}{\text{s}}, \quad \phi_0 = 4,$$
$$\phi_1 = 20, \quad \phi_2 = 3, \quad \text{and} \quad \phi_3 = 9,$$

where the value of 0.5 given to $\sigma$ translates into a geometric average between the minimum-CFL-condition-based pseudo-time step and the maximum-CFL-condition-based pseudo-time step. The convergence threshold $\xi_{\text{verge}}$ is low enough that a decrease in $\xi_{\text{verge}}$ would not result in any noticeable difference of the pressure contours shown in Fig. 6. It is noted that the use of the entropy correction by Yee *et al.* [19] with $\tilde{\zeta} = 0.2$ is here used to avoid a carbuncle phenomenon near the blunt leading edge.

Table I shows the CPU time and effective iterations needed to reach convergence for the marching-window, marching-window/multizone, active-domain, multizone, and standard cycles. Due to the CFL = 1 restriction on the traveling speed of the waves in the flow field to approximately one grid line per iteration, the standard cycle requires a number of iterations proportional to the number of grid lines along the streamwise direction, i.e.,

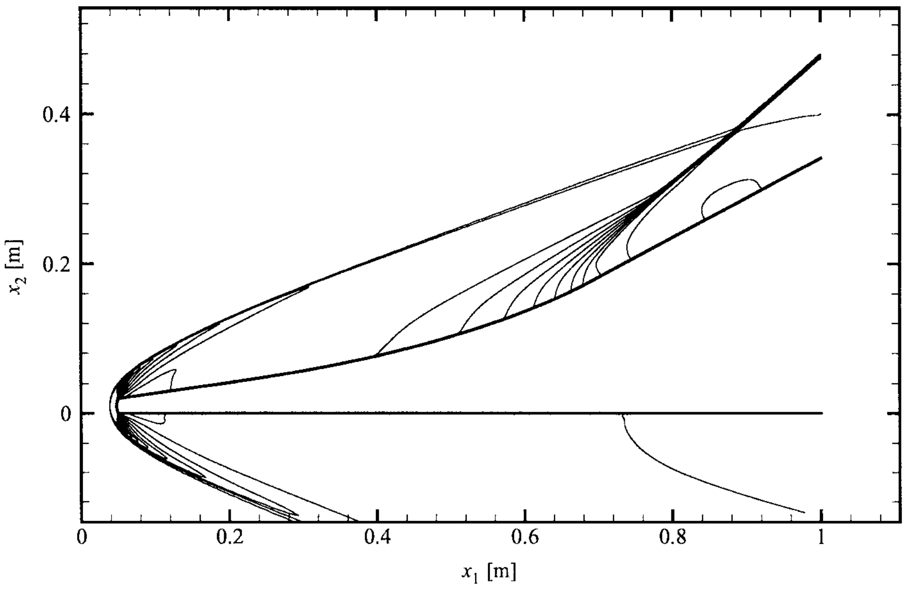

**FIG. 6.** Pressure contours for the blunt leading edge inviscid supersonic inlet case obtained using a $512 \times 256$ grid. The inflow conditions correspond to $M = 5$, $P = 4$ kPa, and $T = 240$ K. No difference is noticeable between the pressure contours obtained with the different cycles.

**TABLE I**
**Effective Iteration Count, Work, and Storage Comparison at a CFL Number of Unity, for the Blunt Leading Edge Inviscid Supersonic Inlet Case**

| | 128 × 64 nodes | | | 512 × 256 nodes | | |
|---|---|---|---|---|---|---|
| Cycle | Iter. | Work | Stor. | Iter. | Work | Stor. |
| Marching window/multizone | 34.4 | 1.0 | 0.11 | 44.9 | 22.2 | 0.91 |
| Marching window | 40.4 | 1.1 | 0.11 | 86.8 | 38.9 | 0.91 |
| Active domain | 55.6 | 1.3 | 0.10 | 102.2 | 39.2 | 0.81 |
| Multizone | 158.5 | 3.8 | 1.0 | 318.6 | 131.6 | 16.0 |
| Standard cycle | 293.0 | 6.5 | 1.0 | 1391.0 | 524.3 | 16.0 |

293 iterations for a 128 × 64 mesh to 1391 iterations for a 512 × 256 mesh. The multizone cycle suffers the same symptoms but has the extra advantage of *not* allocating work to the zones where all nodes exhibit a $\xi$ smaller than the user-specified threshold, thereby reducing the computing to a smaller and smaller domain as the iteration count progresses and the nonconverged flow region moves toward the domain exit. This results in the impressive savings in iteration count of 1.8 times for the coarse mesh and of 4.4 times for the fine mesh. Both the active-domain cycle and the marching-window cycle decrease further the iteration count by allowing a computational window to travel in space following the propagation of the waves. This results in a decrease in effective iterations, compared to the standard cycle using the 512 × 256 mesh, of 14 and 16 times for the active domain and marching window, respectively. Furthermore, the use of multizone decomposition inside the marching window focuses the pseudo-time stepping effort to the regions requiring more iterations to reach convergence, such as the region of subsonic flow upstream of the inlet blunt leading edge, hence resulting in only 45 effective iterations to reach convergence and an overall reduction in effective iterations of 31 times compared to the standard cycle.

Remember that the standard cycle, the multizone cycle, and the marching-window cycle (with and without multizone decomposition) all guarantee that

$$\xi \leq \xi_{\text{verge}} \ \forall \quad \text{inner nodes,}$$

once convergence is reached [as previously stated in Eq. (33)], which is a necessary condition for a well-posed acceleration technique. The latter is *not* a property of the active-domain method when the discretization stencil for the streamwise convection derivative depends on downstream nodes in locally supersonic flow. The Yee TVD limiter used here has this property, and the active-domain algorithm induces a converged residual that does not satisfy the convergence criterion of Eq. (33).

It could be argued that raising the CFL number would improve the standard cycle over the others for this particular case. The effect of a change of CFL number was not investigated but is addressed in the subsequent test problems. It is noted that for many realistic problems dominated by nonlinear phenomena, nonlinear stability conditions, restrict the use of high CFL numbers until the waves have started to settle down considerably.

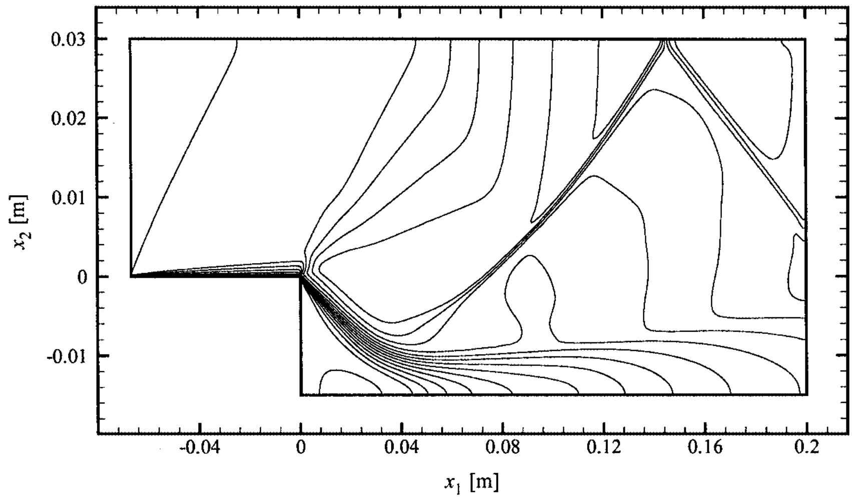

**FIG. 7.** Mach number contours of the backward-facing step case, obtained using a 256 × 128 node mesh. The inflow conditions are set to a Mach number of 2, a Reynolds number per meter of 5 million, and a temperature of 300 K.

## 7.2. *Backward-Facing Step*

Although the gains in computing efficiency obtained through the marching-window cycle might be expected for an entirely supersonic problem without any reverse-flow regions, we proceed to show in this section that the marching window can reduce convergence time considerably even when a substantial portion of the flow field is separated.

Air enters the computational domain at a Mach number of 2, a Reynolds number per meter of 5 million, and a temperature of 300 K. Symmetry conditions are applied at $x_2 = -0.015$ m and at $x_2 = 0.03$ m, inflow at $x_1 = -0.067$ m, and outflow at $x_1 = 0.2$ m, and an adiabatic wall boundary condition is in effect elsewhere. The Mach number contours shown in Fig. 7 show the limits of the recirculation region ($0 \leq x_1 \leq 0.04$) in which 25% of the grid lines along $X_1$ and 50% of the grid lines along $X_2$ are placed. The investigation is performed using a 256 × 128 mesh, clustered at the surfaces; it is noted that no significant difference in the trends is observed using a coarser mesh of 128 × 64 nodes.

A minimum/maximum CFL averaged local time step as specified in Eq. (30) is used for all cycles, and the convergence threshold along with the other user-defined constants are specified to

$$\sigma = 0.5, \quad \xi_{\text{verge}} = 100\,\frac{1}{\text{s}}, \quad \varphi_{\text{verge}} = 5000\,\frac{1}{\text{s}}, \quad \phi_1 = 20, \quad \phi_2 = 3, \quad \text{and} \quad \phi_3 = 9.$$

When a variable CFL number is used, it is set to a function of $\xi_{\max}$ as opposed to the iteration count to enable a more adequate comparison between the different cycles, since the convergence history has a different dependence on the iteration count for each cycle. Note that $\xi_{\max}$ stands for the maximum value of $\xi$ in the computational window, which corresponds to the entire domain for the multizone and standard cycles. In this case, a $\xi_{\max}$ varying between $10^5\,\frac{1}{\text{s}}$ and $10^3\,\frac{1}{\text{s}}$ is made to induce a CFL number varying between 1 and 10. The variation in CFL number is necessary due to nonlinear stability restrictions on the time step size, and it is assumed that an inverse relationship exists between $\xi_{\max}$ and the maximum allowable CFL number for stable and predictable convergence.

**TABLE II**
**Effective Iteration Count, Work, and Storage Comparison for the Backward-Facing Step Case for a 256 × 128 Grid**

| | CFL = 1 | | | $1 \leq$ CFL $\leq 10$ | | |
|---|---|---|---|---|---|---|
| Cycle | Iter. | Work | Stor. | Iter. | Work | Stor. |
| Marching window/multizone | 459 | 2.0 | 1.0 | 229 | 1.0 | 1.0 |
| Marching window | 1166 | 3.9 | 1.0 | 355 | 1.2 | 1.0 |
| Multizone | 1259 | 5.9 | 4.3 | 698 | 3.2 | 4.3 |
| Standard cycle | 3164 | 10.4 | 4.3 | 1054 | 3.5 | 4.3 |

The influence of a change in CFL number on the efficiency of the different cycles can be seen in Table II. The marching-window/multizone cycle at a CFL of unity requires 6.9 times fewer iterations to reach convergence than the standard cycle. Yet, if the CFL is varied between 1 and 10 the speedup is reduced to 4.6 times. As expected, a rise in the CFL number greatly helps the propagation of the waves along the streamwise direction for the standard cycle, while the transmission of the streamwise information is already adequate at a CFL of unity for the marching-window and marching-window/multizone cycles. Nevertheless, the recirculation region is the iteration bottleneck of this problem and the number of steps necessary to solve it is similar for all approaches. For the standard cycle, since the entire computational domain is computed at every step, a region of slow convergence somewhere in the flow field translates in a very high number of effective iterations, whether the region of slow convergence is very small or not. On the other hand, the marching-window algorithm focuses the effort on the region of slow convergence, consequently resulting in much improved algorithm efficiency.

## 7.3. *Concatenated Channels: Shock/Boundary Layer Interactions*

The ability of the marching-window algorithm to solve shock/boundary layer interactions at hypersonic flow conditions is now tested. The geometry involves the concatenation of a $1.0 \times 0.5$ m channel to a $0.69 \times 0.38$ m channel through a $37^\circ$ compression ramp. Air enters the first channel at uniform conditions of $M = 5$, $P = 1000$ Pa, and $T = 450$ K. Fixed temperature ($T_{\text{wall}} = 450$ K) wall boundary conditions are applied on bottom and top boundaries, with a grid clustered at both walls. As for the backward-facing step case, a geometric averaged local time step is utilized to enhance wave propagation through high aspect ratio cells, while the other user-adjustable parameters are set to

$$\sigma = 0.5, \quad \xi_{\text{verge}} = 100\frac{1}{\text{s}}, \quad \varphi_{\text{verge}} = 5000\frac{1}{\text{s}}, \quad \phi_1 = 20, \quad \phi_2 = 3, \quad \text{and} \quad \phi_3 = 9.$$

From the effective pressure contours of Fig. 8, two recirculation regions are visible: one at the start of the shock formed by the $37^\circ$ wedge, and one at the point where the shock impinges on the top wall boundary layer. Both recirculation zones are of appreciable size due to the very low Reynolds number of the flow, which helps generate thick incoming boundary layers. The major obstacle in converging this flow field efficiently comes from the high difference in time scales between the convection-dominated flow in the middle of

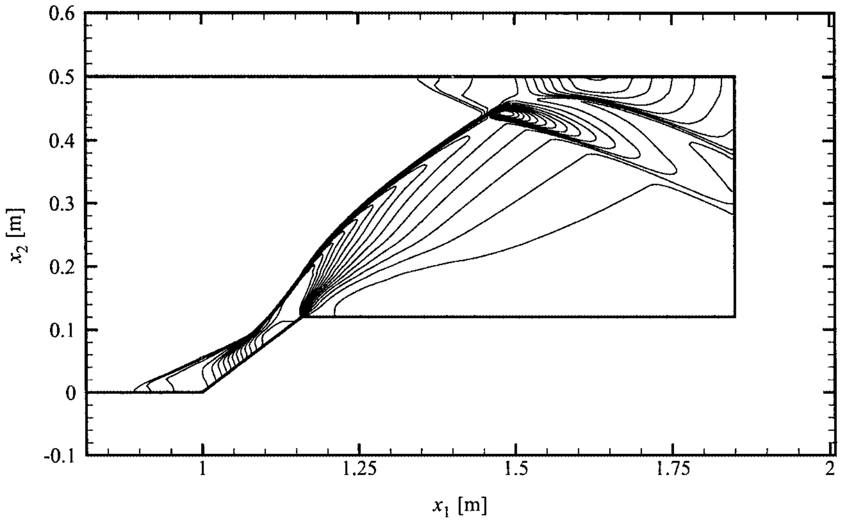

**FIG. 8.** Effective pressure contours of the concatenated channels case, obtained using a 512 × 256 mesh. Air enters the first channel at a Mach number of 5, a pressure of 1000 Pa, and a temperature of 450 K.

the channels and the viscous-dominated recirculation zones. Time-accurate simulations of a similar problem indicate that the amount of time required for the separated flow regions to reach steady state is typically one order of magnitude more than the time needed for the shock structure to establish itself. Consequently, one would prefer high pseudo-time steps to be used in the recirculation zones for fast convergence, but unfortunately the step size is limited by nonlinear stability restrictions, which are of importance especially near the nonconverged shock waves. For these reasons, it is not surprising that so many iterations are needed for the standard cycle to reach convergence, as Table III shows: 4547 iterations for a CFL number varying between 0.1 and 1 and 2342 iterations for the range $0.1 \leq \text{CFL} \leq 10$. Similarly to the backward-facing step, the CFL number is linked to $\xi_{\max}$ such that at $\xi_{\max} = 10^4 \frac{1}{s}$, the CFL number is 10, and at $\xi_{\max} = 10^6 \frac{1}{s}$, the CFL number is 0.1.

The marching-window/multizone cycle performs particularly well, as the work is focused on the reverse-flow regions, while the rest of the domain is quasihyperbolic/parabolic and needs only a small amount of work to reach convergence (see Fig. 9). The use of the

**TABLE III**
**Effective Iteration Count, Work, and Storage Comparison for the Concatenated Channels Test Case**

| | $0.1 \leq \text{CFL} \leq 1$ | | | $0.1 \leq \text{CFL} \leq 10$ | | |
|---|---|---|---|---|---|---|
| Cycle | Iter. | Work | Stor. | Iter. | Work | Stor. |
| Marching window/multizone | 431 | 1.9 | 1.0 | 219 | 1.0 | 1.0 |
| Marching window | 1111 | 4.0 | 1.0 | 444 | 1.7 | 1.0 |
| Multizone | 1571 | 7.8 | 6.2 | 1215 | 5.9 | 6.2 |
| Standard cycle | 4547 | 15.5 | 6.2 | 2342 | 8.0 | 6.2 |

*Note*. The mesh size is 256 × 128 nodes.

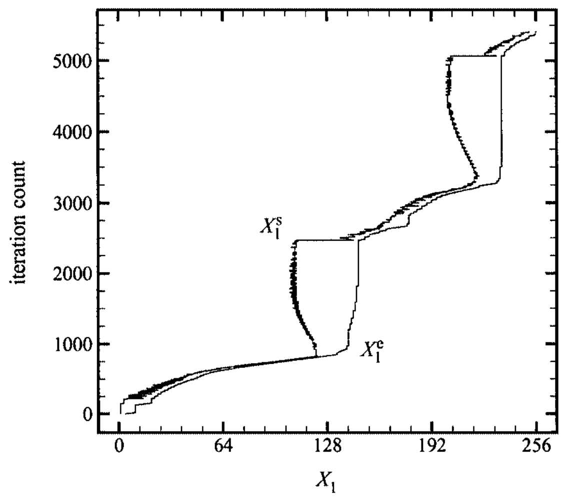

**FIG. 9.** Location of the marching-window upstream and downstream boundaries for the concatenated channels case using the marching-window/multizone cycle, with a variable CFL number, i.e., $0.1 \leq \text{CFL} \leq 10$. Notice the high amount of work spent on the recirculation zones in the vicinity of $X_1 \sim 128$ and $X_1 \sim 218$, while very few steps are needed to converge the quasihyperbolic/parabolic regions.

marching window coupled with a multizone strategy makes possible a decrease in effective iterations of eight times compared to the standard cycle, independently of the CFL number used, as shown in Table III.

There might be doubts as to the adequacy of a varying CFL number function of $\xi_{\text{max}}$ as a means of comparing different cycles. For this reason, additional simulations involving the marching-window/multizone and standard cycles are performed in which the CFL number is made a function of the iteration count and raised to 10 as rapidly as the stability conditions permit. Some 1930 iterations for the standard cycle are needed for convergence while 197 iterations are needed for the marching-window/multizone cycle. Again, while a slight increase in efficiency for the standard cycle is apparent, approximately the same amount is noticeable for the marching-window/multizone cycle.

As in the previous test cases, a convergence criterion of $\xi_{\text{verge}} = 100\frac{1}{\text{s}}$ is found necessary to obtain reasonable accuracy, and no discernible difference is observed between the contours of properties obtained with the different cycles. Even if both the marching-window and the standard cycle guarantee the convergence criterion of Eq. (33) will be satisfied once convergence is attained, the governing equations have multiple roots due to their nonlinearity, and a different flow solution could be obtained by the different cycle strategies. For all test cases presented here, however, it is verified that the same root is obtained independently of the acceleration technique.

The sensitivity of the user-adjustable parameters for the marching-window cycle is assessed for this test case in Table IV. It is seen that the performance of the marching window is not affected considerably by a change in the average zone length $\phi_1$ or by a change in $\phi_2$, the latter being the number of iterations before a reading of the streamwise ellipticity sensor $\xi$ is taken. For $\phi_1$ varied from 10 to 40, the number of effective iterations is observed to change by only 12%, and for $\phi_2$ varied from 1 to 15, the number of effective iterations increases by 14%. On the other hand, the parameters $\phi_3$ and $\varphi_{\text{verge}}$ are seen to affect the performance of the algorithm considerably. Raising $\phi_3$ from 9 to 36 increases twofold the

**TABLE IV**
**Sensitivity of the Effective Iteration Count and Work to the User-Defined Constants for the Concatenated Channels Test Case**

| $\phi_1$ | $\phi_2$ | $\phi_3$ | $\varphi_{\text{verge}}$, $\frac{1}{\text{s}}$ | Work | Iter. |
|---|---|---|---|---|---|
| 20 | 3 | 9 | $5 \times 10^3$ | 1.00 | 1.0 |
| 10 | 3 | 9 | $5 \times 10^3$ | 1.20 | 1.12 |
| 40 | 3 | 9 | $5 \times 10^3$ | 1.00 | 1.02 |
| 20 | 15 | 9 | $5 \times 10^3$ | 1.15 | 1.12 |
| 20 | 1 | 9 | $5 \times 10^3$ | 0.96 | 0.98 |
| 20 | 3 | 5 | $5 \times 10^3$ | 1.46 | 1.41 |
| 20 | 3 | 18 | $5 \times 10^3$ | 1.19 | 1.24 |
| 20 | 3 | 36 | $5 \times 10^3$ | 1.89 | 1.99 |
| 20 | 3 | 9 | $5 \times 10^2$ | 1.33 | 1.40 |
| 20 | 3 | 9 | $5 \times 10^4$ | 1.57 | 1.54 |

*Note*. The marching-window/multizone cycle is used with a mesh size of $256 \times 128$ nodes and a CFL range of $0.1 \leq \text{CFL} \leq 10$.

number of effective iterations, and increasing $\varphi_{\text{verge}}$ tenfold results in an increase of 54% in the effective iterations count. The high sensitivity of the effective iterations on either $\phi_3$ or $\varphi_{\text{verge}}$ is due to the high dependence of the width of the marching window on these parameters. When the marching window encloses too tightly a zone of streamwise ellipticity, the solution needs to be converged locally several times, hence increasing the work. When the marching window overestimates the size of a streamwise elliptic region, the high number of iterations needed locally to converge a streamwise elliptic region is spent on a larger portion of the computational domain, hence resulting in decreased performance.

## 7.4. *Time-Accurate Simulation of an Exploding Cavity in a Supersonic Stream*

The performance of the different cycles on a time-accurate simulation of a stagnant high-pressure flow pocket exploding into a Mach 2 air stream is investigated in this section. The computational domain has dimensions as shown in Fig. 10 and is spanned by a grid

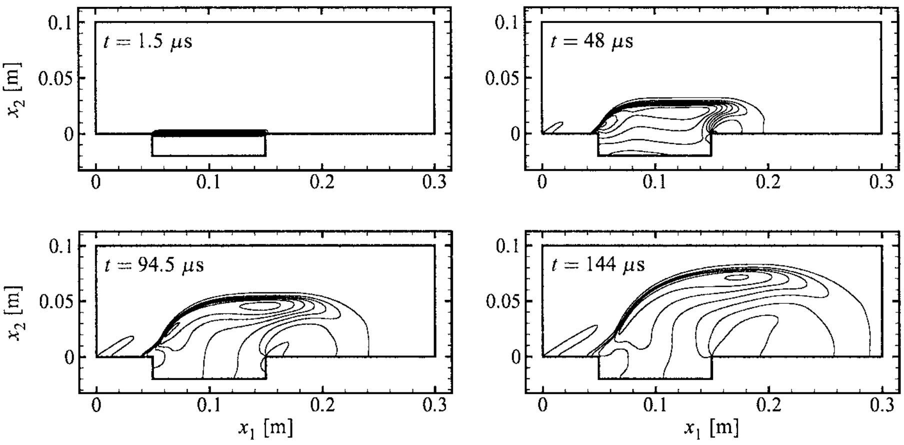

**FIG. 10.** Effective pressure contours of the exploding cavity case using a mesh composed of $256 \times 128$ nodes, and a time step of 1.5 μs. At $t = 0$, the flow outside the cavity is air uniformly distributed at $P = 10\,\text{kPa}$, $T = 300\,\text{K}$, and $M = 2$, while the air inside the cavity is set to $P = 100$ kPa, $T = 2000$ K, and $M = 0$.

**TABLE V**
**Effective Iteration Count, Work, and Storage Comparison for the Time-Accurate Simulation of an Exploding Cavity**

| Cycle | Iterations | Work | Storage |
|---|---|---|---|
| Multizone | 1505 | 1.0 | 5.0 |
| Sweeping window/multizone | 2034 | 1.5 | 1.0 |
| Sweeping window | 2980 | 1.9 | 1.0 |
| Standard cycle | 4418 | 2.0 | 5.0 |

*Note.* The CFL number is unity.

composed of $256 \times 128$ nodes, of which $110 \times 38$ are allocated to the cavity. Inflow, outflow, and symmetry conditions are applied to the left, right, and top boundaries, respectively, while an adiabatic wall condition is in effect elsewhere. The mesh is clustered at the inflow and at all surfaces to capture the turbulent boundary layer correctly and is not altered in time. The flow field at $t = 0$ for $x_2 \geq 0$ is set to a pressure of 10 kPa, a temperature of 300 K, and a flow Mach number of 2, while for $x_2 < 0$, the pressure, temperature, and Mach number are set to 100 kPa, 2000 K, and 0, respectively. The solution is iterated in pseudotime starting from the converged solution of the previous physical time step until the maximum value $\xi$ in the computational domain falls below $\xi_{\text{verge}}$. The physical time step, $\Delta t$, is fixed to 1.5 $\mu$s, while the following user-defined parameters are in use:

$$\sigma = 0.3, \quad \xi_{\text{verge}} = 100\frac{1}{\text{s}}, \quad \varphi_{\text{verge}} = 5000\frac{1}{\text{s}}, \quad \phi_1 = 20, \quad \phi_2 = 3, \quad \text{and} \quad \phi_3 = 9.$$

Due to the strict convergence criterion utilized and the use of the same residual, all acceleration techniques result in the same answer at all time steps despite noticeable differences in CPU work, as Table V attests: a twofold decrease in work is achieved through the use of the multizone cycle, while the sweeping-window/multizone cycle decreases the work by one-quarter. The performance of the sweeping window is not particularly good for this problem, as the wave-propagation direction is more toward time than in the stream-wise coordinate due to the relatively small physical time step. Furthermore, since only 20 effective iterations are needed on average per time level, the overhead work induced by sweeping becomes more important, as the greater discrepancy observed for this case between the reduction in effective iterations and CPU work shows. There is, nonetheless, a nonnegligible fivefold reduction in storage when using the sweeping window.

## 8. SUMMARY AND CONCLUSIONS

A novel acceleration technique is presented which is aimed at accelerating the convergence of the Favre-averaged Navier–Stokes equations in the supersonic/hypersonic regime for flow fields with large streamwise separated flow regions. Similarly to the active-domain method [13], the marching window iterates in pseudotime a bandlike computational domain of minimal width which adjusts to the size of the streamwise elliptic regions when encountered. However, contrarily to the active-domain method, it is shown that the marching window guarantees the residual on all nodes to be below a user-defined threshold when convergence is reached and, hence, results in the same converged solution (within the tolerance of the convergence criterion) as the one obtained by standard pseudo-time-marching methods. Further, a streamwise ellipticity sensor based on the Vigneron splitting [1] is developed which ensures that the downstream boundary of the marching window advances sufficiently so that regions of significant streamwise ellipticity are contained within the marching-window subdomain. It is noted that while the Vigneron splitting sensor does not capture all possible streamwise elliptic phenomena, this does not affect the accuracy of the final solution and only affects the performance of the marching window as an acceleration technique. Also, a multizone decomposition is implemented inside the marching window to restrict the computing to the zones where the residual is above the user-defined convergence threshold. This is shown to further decrease the work needed for convergence by close to two times for the problems shown herein.

The use of the marching window with multizone decomposition on a backward-facing step and a shock boundary layer interaction flow field (where one or several large streamwise separated regions are present) reveals a four- to sixfold decrease in storage and a four- to eightfold decrease in work compared to the standard cycle. The proposed algorithm is also shown to work well at a low CFL number in regions of quasihyperbolicity/parabolicity and is recommended for stiff problems with high nonlinear stability restrictions on the time-step size. A variant of the marching window designed for time-accurate simulations is observed to result in a fivefold reduction in storage and 25% reduction in work for the time-accurate exploding cavity case investigated herein. The reduction in computational work through the use of the marching window is made possible by limiting the high number of iterations needed to converge the streamwise separated regions to the region in question. The amount of storage needed is also significantly reduced if no memory is allocated to the nodes outside of the marching-window subdomain.

The marching window does not impose any restriction on the discretization-stencils part of the residual or on the pseudo-time-stepping method. While not implemented here, the numerous acceleration techniques available for pseudo-time stepping (such as multigrid, block relaxation [31], preconditioning, Newton–Krylov, etc.) can be used in conjunction with the marching window. Furthermore, the marching window is not limited to the Favre-averaged Navier–Stokes equations and its extension to other governing equations would only require a redefinition of the ellipticity sensor shown in Eq. (39).

The performance of the algorithm is seen to be sensitive to the user-defined ellipticity threshold constant $\varphi_{\text{verge}}$ and the marching window minimal width $\phi_3$. It is unclear at this stage by how much these parameters would need to be altered for very different flow properties and physical domain sizes. The dependency on the problem setup seems not too severe, as the same values for the user-specified constants are used for all cases shown in this paper.

## ACKNOWLEDGMENTS

This work has been supported by the Natural Sciences and Engineering Research Council (NSERC). The first author would like to thank all writers of free software for donating the compilers, typesetting software, vector drawing programs, and so forth, that were used in this study. The help of G. Fusina in proofreading the manuscript is greatly appreciated. The authors are also grateful for the constructive criticisms of the reviewers.

## REFERENCES

1. Y. C. Vigneron, J. V. Rakich, and J. C. Tannehill, *Calculations of Supersonic Viscous Flow over Delta Wings with Sharp Subsonic Leading Edges*, AIAA Paper 78-1137 (1978).

2. S. L. Lawrence and J. C. Tannehill, *An Upwind Algorithm for the Parabolized Navier–Stokes Equations*, AIAA Paper 86-1117 (1986).
3. J. Tannehill, J. O. Ievalts, and S. Lawrence, *An Upwind Parabolized Navier–Stokes Code for Real Gas Flows*, AIAA Paper 89-0713 (1988).
4. D. D'Ambrosio and R. Marsilio, A numerical method for solving the parabolized three-dimensional Navier–Stokes equations, *Comp. Fluids* **26**, 587 (1997).
5. J. H. Miller, J. C. Tannehill, S. L. Lawrence, and T. A. Edwards, Parabolized Navier–Stokes code for hypersonic flows in thermo-chemical equilibrium or nonequilibrium, *Comp. Fluids* **27**, 199 (1998).
6. J. H. Miller, J. C. Tannehill, and S. L. Lawrence, Parabolized Navier–Stokes algorithm for solving supersonic flows with upstream influences, *AIAA J.* **38**, 1837 (2000).
7. G. D. Power and T. J. Barber, Analysis of complex hypersonic flows with strong viscous/inviscid interaction, *AIAA J.* **26**, 832 (1988).
8. C.-L. Chang and C. L. Merkle, The relation between flux vector splitting and parabolized schemes, *J. Comput. Phys.* **80**, 344 (1989).
9. N. K. Yamaleev and J. Ballmann, Iterative space-marching method for compressible sub-, trans-, and supersonic flows, *AIAA J.* **38**, 225 (2000).
10. S. G. Rubin and D. R. Reddy, Analysis of global pressure relaxation for flows with strong interaction and separation, *Comp. Fluids* **11**, 281 (1983).
11. M. Barnett and R. T. Davis, Calculation of supersonic flows with strong viscous-inviscid interaction, *AIAA J.* **24**, 1949 (1986).
12. J. E. Bardina, Three-dimensional Navier–Stokes method with two-equation turbulence models for efficient numerical simulation of hypersonic flows, *presented at the 30th Joint Propulsion Conference and Exhibit, Indianapolis, IN*, AIAA Paper 94-2950 (1994).
13. K. Nakahashi and E. Saitoh, Space-marching method on unstructured grid for supersonic flows with embedded subsonic regions, *AIAA J.* **35**, 1280 (1997).
14. H. Morino and K. Nakahashi, *Space-Marching Method on Unstructured Hybrid Grid for Supersonic Viscous Flows*, AIAA Paper 99-0661 (1999).
15. D. C. Wilcox, Reassessment of the scale determining equation for advanced turbulence models, *AIAA J.* **26**, 1299 (1988).
16. H. Viviand, *Conservative Forms of Gas Dynamics Equations* (La Recherche Aérospatiale) (1974).
17. M. Vinokur, Conservative equations of gas-dynamics in curvilinear coordinate systems, *J. Comput. Phys.* **14**, 105 (1974).
18. P. L. Roe, Approximate Riemann solvers, parameter vectors, and difference schemes, *J. Comput. Phys.* **43**, 357 (1981).
19. H. C. Yee, G. H. Klopfer, and J.-L. Montagné, High-resolution shock-capturing schemes for inviscid and viscous hypersonic flows, *J. Comput. Phys.* **88**, 31 (1990).
20. P. Batten, M. A. Leschziner, and U. C. Goldberg, Average-state Jacobians and implicit methods for compressible viscous and turbulent flows, *J. Comput. Phys.* **137**, 38 (1997).
21. B. Parent and J. P. Sislian, Turbulent hypervelocity fuel/air mixing by cantilevered ramp injectors, *presented at the 10th AIAA International Aerospace Planes Hypersonics Technologies Conference, Kyoto, Japan*, AIAA Paper 2001-1888 (2001).
22. W. R. Briley and H. McDonald, Solution of the multidimensional compressible Navier–Stokes equations by a generalized implicit method, *J. Comput. Phys.* **24**, 372 (1977).
23. R. Beam and R. F. Warming, An implicit factored scheme for the compressible Navier–Stokes equations, *AIAA J.* **16**, 393 (1978).
24. S. V. Patankar, *Numerical Heat Transfer and Fluid Flow*, Taylor and Francis, 1980.
25. R. W. MacCormack, Iterative modified approximate factorization, *Comp. & Fluids* **30**, 917 (2001).
26. S. Yoon and A. Jameson, Lower-Upper implicit scheme for high-speed inlet analysis, *AIAA J.* **25**, 1052 (1987).
27. D. C. Wilcox, *Turbulence Modeling for CFD*, DCW Industries, 1994.

28. C. B. Laney, *Computational Gasdynamics*, Cambridge University Press, Cambridge, New York, NY, 1998.
29. M. L. Sawley and J. K. Tegner, A data parallel approach to multi-block flow computations, *Int. J. Numer. Methods Fluids* **19**, 707 (1994).
30. M. Rosenfeld, The alternating direction multi-zone implicit method, *J. Comput. Phys.* **110**, 212 (1994).
31. C. de Nicola, R. Tognaccini, and V. Puoti, Local block relaxation method for the solution of equations of gasdynamics, *AIAA J.* **38**, 1377 (2000).