ELSEVIER

Comput. Methods Appl. Mech. Engrg. 123 (1995) 161–172

Computer methods in applied mechanics and engineering

Sensitivity-based methods for convergence acceleration of iterative algorithms

Kwang-Yoon Choi[1], George S. Dulikravich*

Department of Aerospace Engineering, The Pennsylvania State University, University Park, PA 16802, USA

Received 17 March 1994

Abstract

A new method to accelerate the convergence of iterative schemes for the numerical integration of systems of partial differential equations has been developed. The basic idea is that the residual at a grid point depends on the values of the solution vector at the neighboring grid points used in the local discretization approximation. Thus, the new acceleration method is based on the sensitivity of the future residual to the change in the solution vector at the neighboring grid points with the objective to minimize the future residual. The result is a set of optimum iterative relaxation parameters for the entire flow field or for each individual grid line. The method is easy to implement in the existing codes. We have applied it to a finite difference code for two-dimensional incompressible Navier–Stokes equations. Test cases involve laminar and turbulent flows with severe grid clustering and flow separation. The results are compared with those of a basic explicit Runge–Kutta (RK) time-stepping iterative algorithm and with the implicit residual smoothing (IRS) and the distributed minimal residual (DMR) acceleration techniques. The new acceleration scheme is shown to be superior to these methods especially on highly-clustered grids.

1. Introduction

One of the consistent goals in computational fluid dynamics is to improve the efficiency of numerical techniques by reducing the total computing time required by an iterative algorithm. One of the successful, explicit techniques used to integrate Euler and Navier–Stokes equations is the Runge–Kutta (RK) explicit time-stepping algorithm [1]. Several attempts have been made to accelerate the convergence of this method. Some of the more successful acceleration methods are based on local time stepping [1], implicit residual smoothing [1, 2], enthalpy damping [1], multi-griding [2] and preconditioning [3].

Here, we plan to elaborate on an optimum extrapolation technique in order to find optimum relaxation parameters for iterative algorithms. An extrapolation procedure based on the power method and the minimal residual method (MRM) was successfully applied [4] to the multi-grid algorithm. The MRM uses equal optimal weights for the corrections to every equation in a system of partial differential equations that is solved, but it has not been shown to accelerate the scheme without multi-griding. The distributed minimal residual (DMR) method [5–8] based on the general non-linear minimal residual (GNLMR) method [9, 10] allows each component of the solution vector in a system of equations to have its own sequence of optimized relaxation parameters. The DMR method was found to be capable of reducing the computation time by 10–75% depending on the test case and grid used. It is an improvement over a well-known general minimal residual (GMRES) method [11] which requires a

* Corresponding author.

[1] Present address: Samsung Aerospace Industries Ltd, Taejon, South Korea.

SSDI 0045-7825(94)00776-4

large amount of computer memory. All these optimum extrapolation methods belong to the general class of Krylov subspace methods.

A new convergence acceleration algorithm based on the sensitivity of future residual to flow variables at the surrounding grid points is presented in this paper [12–16]. The new method was tested on uniform and highly-clustered computational grids and for both low and high Reynolds number flows including flows with separation. It was found to perform consistently better than the IRS and the DMR acceleration methods.

2. Incompressible Navier–Stokes equations

The two-dimensional, incompressible Navier–Stokes equations with Chorin's artificial compressibility method [17] can be written in conservative vector form as

$$\frac{\partial \boldsymbol{Q}}{\partial \tau}+\frac{\partial \boldsymbol{E}}{\partial \xi}+\frac{\partial \boldsymbol{F}}{\partial \eta}-\boldsymbol{H}+\boldsymbol{S}=0 \tag{1}$$

where τ is the time, $\boldsymbol{Q}$ is the solution vector, $\boldsymbol{E}$ and $\boldsymbol{F}$ are the inviscid flux vectors, $\boldsymbol{H}$ is the vector of viscous terms and $\boldsymbol{S}$ is the vector of source terms. Thus

$$\boldsymbol{Q}=\frac{1}{J}\begin{Bmatrix} p \\ \beta \\ u \\ v \end{Bmatrix} \qquad \boldsymbol{E}=\frac{1}{J}\begin{Bmatrix} U \\ Uu+\xi_x p \\ Uv+\xi_y p \end{Bmatrix} \qquad \boldsymbol{F}=\frac{1}{J}\begin{Bmatrix} V \\ Vu+\eta_x p \\ Vv+\eta_y p \end{Bmatrix} \tag{2}$$

Here, U and V are the contravariant velocity components, u and v are Cartesian velocity components, p is the pressure, ξ and η are the curvilinear, non-orthogonal, boundary conforming grid coordinates, J is the determinant of the Jacobian transformation matrix, $J=\det(\partial(\xi,\eta)/\partial(x,y))$, and β is the user-specified artificial compressibility parameter [17]. The term $\partial(p/J\beta)/\partial\tau$ was added to the mass conservation to make the entire system non-singular. Adding the non-physical term, $\partial(p/J\beta)/\partial\tau$, does not affect the steady-state solution since all the time derivative terms vanish as the solution converges to a steady-state.

3. Implicit residual smoothing (IRS) method [2]

Residual smoothing permits the use of higher values of Courant–Friedrichs–Lewy (CFL) number thus accelerating the convergence. The implicit form of the residual smoothing for a two-dimensional case can be written as

$$[1-\varepsilon\delta_\xi^2][1-\varepsilon\delta_\eta^2]\bar{R}=R \tag{3}$$

Here, R is the original unsmoothed residual, $\bar{R}$ is the smoothed residual, δ^2 is the second-order differential operator and ε is a user specified coefficient so that

$$\varepsilon \geqslant \frac{1}{4}\left[\left(\frac{\mathrm{CFL}}{\mathrm{CFL}^*}\right)^2-1\right] \tag{4}$$

Here, CFL* is the Courant–Friedrichs–Lewy number for the unsmoothed scheme and CFL is the modified Courant–Friedrichs–Lewy number. The IRS is applied at each stage of the RK scheme. It was found that the computing time was increased by a factor of three if the IRS is applied at every iteration together with the current four-stage RK time-stepping. For the test cases investigated in the paper, the IRS was applied after every 10 iterations for optimum performance.

4. Distributed minimal residual (DMR) method [5–8]

This method extrapolates the solution at iteration level $t+1$ from the previous N iteration levels. Specifically, the DMR is presently formulated as

$$\begin{aligned} Q_1^{n+1} &= Q_1^n + \omega_1^n \delta Q_1^n + \omega_1^{n-1} \delta Q_1^{n-1} + \cdots + \omega_1^{n-(N-1)} \delta Q_1^{n-(N-1)} \\ &\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ Q_M^{n+1} &= Q_M^n + \omega_M^n \delta Q_M^n + \omega_M^{n-1} \delta Q_M^{n-1} + \cdots + \omega_M^{n-(N-1)} \delta Q_M^{n-(N-1)} \end{aligned} \tag{5}$$

Here, ω's are the iterative relaxation parameters (weight factors) to be calculated and optimized, δQ's are the corrections computed with the non-accelerated iteration scheme, N denotes the total number of consecutive iteration steps combined when evaluating the optimum ω's and M stands for the total number of equations in the system that is being iteratively solved. The DMR method calculates optimum ω's to minimize the $L-2$ norm of the future residual of the system integrated over the entire domain. The present formulation of the DMR uses the same values of the $N \times M$ optimized relaxation parameters at every grid point, although different parts of the flow field converge at different rates.

5. Sensitivity-based minimal residual (SBMR) method [12–15]

Conventional iterative algorithms update the flow variables by calculating the amount of corrections without optimizing their influence on the future residual. The SBMR method evaluates the sensitivity of the residual to the solution vectors at surrounding grid points and calculates the optimum amount of corrections necessary to minimize the overall future residual. The basic idea is that the residual at a grid point depends on the values of the solutions vector Q at the neighboring grid points used in the local finite difference approximation. The rate at which the residual, $R_m (m = 1, \ldots, M$: number of equations in the system to be iteratively solved), changes with $Q_s (s = 1, \ldots, S$: number of neighboring grid points involved in the local discretization scheme) is $\partial R_m / \partial Q_s$. These sensitivities can be determined by taking partial derivatives of the finite difference approximation of the residual with respect to the solution vector Q.

Suppose we know the flow variables Q at iteration level t and $t+n$ where n is the number of regular iteration steps performed by the original non-accelerated code. Then, the changes in the components of Q between the two iteration levels in case of the two-dimensional, incompressible Navier–Stokes equations involving mass, x-momentum and y-momentum conservation equations (2) are given as

$$\Delta p_s = (p_s)^{t+n} - (p_s)^t \qquad \Delta u_s = (u_s)^{t+n} - (u_s)^t, \qquad \Delta v_s = (v_s)^{t+n} - (v_s)^t, \tag{6}$$

Using the first two terms of a Taylor series expansion in the artificial time direction, the residual for each of the equations in the system given by Eq. (2) after n iterations will be

$$R_m^{t+n} = R_m^t + \left[\sum_s \frac{\partial R_m^t}{\partial p_s} \Delta p_s \right] + \left[\sum_s \frac{\partial R_m^t}{\partial u_s} \Delta u_s \right] + \left[\sum_s \frac{\partial R_m^t}{\partial v_s} \Delta v_s \right] \tag{7}$$

If we introduce weighting factors α_p, α_u and α_v to corrections Δp, Δu and Δv, respectively, the future solution vector components can be estimated as

$$\begin{aligned} (p_s)^{(t+n)+1} &= (p_s)^t + \alpha_p \, \Delta p_s \\ (u_s)^{(t+n)+1} &= (u_s)^t + \alpha_u \, \Delta u_s \\ (v_s)^{(t+n)+1} &= (v_s)^t + \alpha_v \, \Delta v_s \end{aligned} \tag{8}$$

Subsequently, the future residual at $(t+n)+1$ can be approximated by

$$R_m^{(t+n)+1} = R_m^t + \left[\sum_s \frac{\partial R_m^t}{\partial p_s} (\Delta p_s \, \alpha_p) \right] + \left[\sum_s \frac{\partial R_m^t}{\partial u_s} (\Delta u_s \, \alpha_u) \right] + \left[\sum_s \frac{\partial R_m^t}{\partial v_s} (\Delta v_s \, \alpha_v) \right] \tag{9}$$

For now, each of the α's is assumed to have the same value over the whole domain, D. Therefore, Eq. (9) can be written as

$$R_m^{(t+n)+1} = R_m^t + \left[\sum_s \frac{\partial R_m^t}{\partial p_s}\Delta p_s\right]\alpha_p + \left[\sum_s \frac{\partial R_m^t}{\partial v_s}\Delta u_s\right]\alpha_u + \left[\sum_s \frac{\partial R_m^t}{\partial v_s}\Delta v_s\right]\alpha_v \tag{10}$$

Subtracting (7) from (10) we have

$$R_m^{(t+n)+1} = R_m^{t+n} + \left[\sum_s \frac{\partial R_m^t}{\partial p_s}\Delta p_s\right](\alpha_p - 1) + \left[\sum_s \frac{\partial R_m^t}{\partial u_s}\Delta u_s\right](\alpha_u - 1) + \left[\sum_s \frac{\partial R_m^t}{\partial v_s}\Delta v_s\right](\alpha_v - 1) \tag{11}$$

The α's are determined such that the L-2 norm of the future residual over the entire domain D will be minimized. In general, this means that

$$\sum_D \left[\frac{\partial\left(\sum_{m=1}^{M}(R_m^{(t+n)+1})^2\right)}{\partial\alpha_q}\right] = 2\sum_D\left[\sum_{m=1}^{M} R_m^{(t+n)+1}\frac{\partial R_m^{(t+n)+1}}{\partial\alpha_q}\right] = 0 \tag{12}$$

where q stands for each primitive flow variable. For simplicity, let us denote the known bracketed terms in Eq. (11) as a_{mp}, a_{mu} and a_{mv} so that

$$R_m^{(t+n)+1} = R_m^{t+n} + a_{mp}(\alpha_p - 1) + a_{mu}(\alpha_u - 1) + a_{mv}(\alpha_v - 1) \tag{13}$$

Substituting (13) into (12) gives the following equations for optimal global α's pertinent to the system given by Eq. (1).

$$\begin{aligned}
&\sum_D\left[\sum_{m=1}^{3}\{R_m^{t+n} + a_{mp}(\alpha_p - 1) + a_{mu}(\alpha_u - 1) + a_{mv}(\alpha_v - 1)\}a_{mp}\right] = 0\\
&\sum_D\left[\sum_{m=1}^{3}\{R_m^{t+n} + a_{mp}(\alpha_p - 1) + a_{mu}(\alpha_u - 1) + a_{mv}(\alpha_v - 1)\}a_{mu}\right] = 0\\
&\sum_D\left[\sum_{m=1}^{3}\{R_m^{t+n} + a_{mp}(\alpha_p - 1) + a_{mu}(\alpha_u - 1) + a_{mv}(\alpha_v - 1)\}a_{mv}\right] = 0
\end{aligned} \tag{14}$$

In Eq. (14), R's and a's are known from the past iteration results. Since each α is assumed to have the same value over the entire computational domain, Eq. (14) gives a system of three simultaneous algebraic equations for α_p, α_u and α_v.

$$\begin{aligned}
&\left[\sum_D\left(\sum_{m=1}^{3} a_{mp}a_{mp}\right)\right](\alpha_p - 1) + \left[\sum_D\left(\sum_{m=1}^{3} a_{mu}a_{mp}\right)\right](\alpha_u - 1) + \left[\sum_D\left(\sum_{m=1}^{3} a_{mv}a_{mp}\right)\right](\alpha_v - 1)\\
&\quad = -\sum_D\left(\sum_{m=1}^{3} R_m^{t+n}a_{mp}\right)\\
&\left[\sum_D\left(\sum_{m=1}^{3} a_{mp}a_{mu}\right)\right](\alpha_p - 1) + \left[\sum_D\left(\sum_{m=1}^{3} a_{mu}a_{mu}\right)\right](\alpha_u - 1) + \left[\sum_D\left(\sum_{m=1}^{3} a_{mv}a_{mu}\right)\right](\alpha_v - 1)\\
&\quad = -\sum_D\left(\sum_{m=1}^{3} R_m^{t+n}a_{mu}\right)\\
&\left[\sum_D\left(\sum_{m=1}^{3} a_{mp}a_{mv}\right)\right](\alpha_p - 1) + \left[\sum_D\left(\sum_{m=1}^{3} a_{mu}a_{mv}\right)\right](\alpha_u - 1) + \left[\sum_D\left(\sum_{m=1}^{3} a_{mv}a_{mv}\right)\right](\alpha_v - 1)\\
&\quad = -\sum_D\left(\sum_{m=1}^{3} R_m^{t+n}a_{mv}\right)
\end{aligned} \tag{15}$$

In the general case of a system (1) composed of M partial differential equations, there will be M constant optimum values of α. This means that the system (15) will become a full $M \times M$ matrix for M unknown optimum α's. Also, each summation in the system (15) which in this example is from $m = 1$ to 3, will become a summation from $m = 1$ to M.

Each application of the SBMR creates significant disturbances in the eigenvalues of the iterative matrix. Thus, the optimum values of α's are determined and applied only periodically followed by a number of iterations with the basic non-accelerated algorithm. This serves as a form of smoothing since too frequent application of the SBMR can lead to instabilities and divergence.

6. Line SBMR (LSBMR) method [15, 16]

The SBMR method calculates the optimum weighting factors α for corrections to the flow variables during the iteration procedure. Since the previous formulation assumes these factors to be constant over the whole computational domain, it should be classified as global SBMR method [12, 14]. For non-uniform grids and rapidly-varying flow variables, optimum α's should not necessarily be the same over the whole computational domain. A modified formulation (line SBMR or LSBMR) will be elaborated upon to allow these α's to have different values [15, 16] from one grid line to another. The formulation will be explained using the two-dimensional, incompressible Navier–Stokes equations.

Let the grid be clustered in the j-direction and let each j = constant grid line have its own set of constant α's. The residual at a point (i, j) incorporates α's at the neighboring grid points plus the point (i, j). For the system given by Eq. (1) this results in

$$
\begin{aligned}
R_m^{(t+n)+1} = R_m^t &+ \sum_{s=i-1}^{i+1} \left[\frac{\partial R_m^t}{\partial p_{s,j-1}} \Delta p_{s,j-1}\, \alpha_p^{j-1} + \frac{\partial R_m^t}{\partial p_{s,j}} \Delta p_{s,j}\, \alpha_p^{j} + \frac{\partial R_m^t}{\partial p_{s,j+1}} \Delta p_{s,j+1}\, \alpha_p^{j+1} \right] \\
&+ \sum_{s=i-1}^{i+1} \left[\frac{\partial R_m^t}{\partial u_{s,j-1}} \Delta u_{s,j-1}\, \alpha_u^{j-1} + \frac{\partial R_m^t}{\partial u_{s,j}} \Delta u_{s,j}\, \alpha_u^{j} + \frac{\partial R_m^t}{\partial u_{s,j+1}} \Delta u_{s,j+1}\, \alpha_u^{j+1} \right] \\
&+ \sum_{s=i-1}^{i+1} \left[\frac{\partial R_m^t}{\partial v_{s,j-1}} \Delta v_{s,j-1}\, \alpha_v^{j-1} + \frac{\partial R_m^t}{\partial v_{s,j}} \Delta v_{s,j}\, \alpha_v^{j} + \frac{\partial R_m^t}{\partial v_{s,j+1}} \Delta v_{s,j+1}\, \alpha_v^{j+1} \right]
\end{aligned} \tag{16}
$$

On each j = constant grid line, three values of constant α's are determined in such a way as to minimize the $L-2$ norm of the future global residual.

$$
\begin{aligned}
2 \sum_D \left(R_1^{t+1} \frac{\partial R_1^{t+1}}{\partial \alpha_p^j} + R_2^{t+1} \frac{\partial R_2^{t+1}}{\partial \alpha_p^j} + R_3^{t+1} \frac{\partial R_3^{t+1}}{\partial \alpha_p^j} \right) = 0 \\
2 \sum_D \left(R_1^{t+1} \frac{\partial R_1^{t+1}}{\partial \alpha_u^j} + R_2^{t+1} \frac{\partial R_2^{t+1}}{\partial \alpha_u^j} + R_3^{t+1} \frac{\partial R_3^{t+1}}{\partial \alpha_u^j} \right) = 0 \\
2 \sum_D \left(R_1^{t+1} \frac{\partial R_1^{t+1}}{\partial \alpha_v^j} + R_2^{t+1} \frac{\partial R_2^{t+1}}{\partial \alpha_v^j} + R_3^{t+1} \frac{\partial R_3^{t+1}}{\partial \alpha_v^j} \right) = 0
\end{aligned} \tag{17}
$$

If $j$max is the total number of j = constant grid lines, then substituting Eq. (16) into (17) results in ($j$max-$j$min + 1) × 3 algebraic equations for the same number of unknown α's. For a given j = constant grid line the three values of α^j appear only in $R_m^{t+1}(i, j-1)$, $R_m^{t+1}(i, j)$ and $R_m^{t+1}(i, j+1)$. The summation over the entire domain in Eq. (17) leaves the terms only with α^{j-2}, α^{j-1}, α^j, α^{j+1} and α^{j+2}. At the solid walls ($j = j\text{min}$ and $j = j\text{max}$) the velocity components are zero and no corrections are needed resulting in $\alpha_u^{j\text{min}} = \alpha_v^{j\text{min}} = \alpha_u^{j\text{max}} = \alpha_v^{j\text{max}} = 0$. For the sake of computational simplicity we used $\alpha_p^{j\text{min}} = \alpha_p^{j\text{min}+1}$ and $\alpha_p^{j\text{max}} = \alpha_p^{j\text{max}-1}$. In this example the simultaneous system of Eqs. (17) yields a block penta-diagonal matrix equation for ($j$max-$j$min + 1) × 3 optimum α's where each block is a 3×3 matrix. In the general case of a two-dimensional system (1) having M partial differential equations, the

block penta-diagonal system (17) will have blocks of $M \times M$ size. In the case of a three-dimensional system (1), the system (17) becomes a block septa-diagonal matrix.

7. Computational results

Computational results are given for a steady, incompressible, viscous flow in a two-dimensional straight channel and U-shaped turn-around channel. Convergence histories of the basic four-stage RK scheme with local time-stepping, and four acceleration methods (IRS, DMR, SBMR and LSBMR) were compared for two different Reynolds numbers with several grid clusterings. For the laminar flow examples, a fully developed parabolic axial velocity (u) profile and zero lateral velocity (v) were specified at the inlet of the channel. In the rest of the flow domain the initial guess for the pressure, v-velocity and u-velocity components were $p = 0, v = 0$ and $u = 1.0 \times 10^{-5}$, respectively. The inlet pressure was iteratively computed by enforcing characteristic boundary conditions at the inlet for both laminar and turbulent flows. At the exit boundary we used non-reflecting [18, 19] boundary conditions since there was a recirculating flow at the exit of the U-shape channel. Both velocity components were set to zero at the channel walls. Wall pressures were computed from the normal momentum equation. No artificial viscosity was used in any of the test cases.

One of the basic objectives of the present study is to develop a method that is capable of accelerating iterative convergence rates on clustered computational grids. For a straight channel, uniform grid spacing was used in the flow direction clustering the grid symmetrically towards lower and upper solid walls. Maximum grid cell aspect ratio ($\Delta x/\Delta y$) occurs on the upper and lower solid boundaries. For different degrees of clustering, the maximum cell aspect ratio on the solid wall was specified and then the grid spacing was increased towards the center of the channel using a geometric progression

$$\Delta y_{j+1} = c\,\Delta y_j \tag{18}$$

The factor c was calculated to satisfy the given maximum cell aspect ratio, $\mathrm{AR}_{\max}$, and the channel height, H. For a U-shaped channel, a clustering function [20] of the following type was used

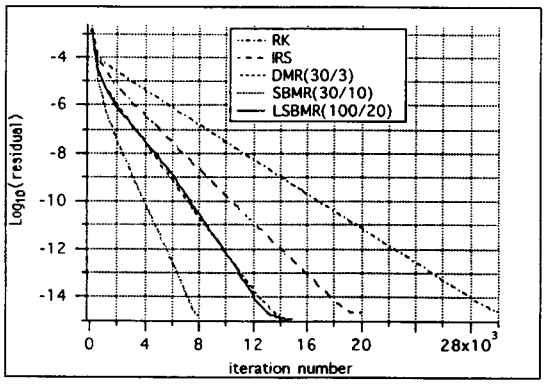
$$y = H\frac{(\kappa + 2\delta)\left[\dfrac{\kappa+1}{\kappa-1}\right]^{(\bar{y}-\delta)/(1-\delta)} - \kappa + 2\delta}{(2\delta+1)\left\{1 + \left[\dfrac{\kappa+1}{\kappa-1}\right]^{\bar{y}-\delta)/(1-\delta)}\right\}} \tag{19}$$

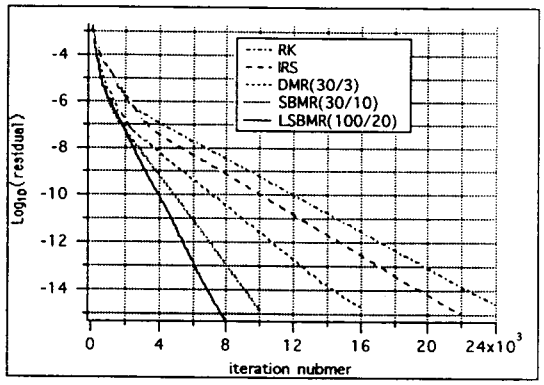
and

$$\bar{x} = \frac{L}{i\text{max-}i\text{min}} \qquad \bar{y} = \frac{H}{j\text{max-}j\text{min}} \tag{20}$$

Here, $\delta = 0.5$ was used to have grid lines symmetrically clustered towards the upper and lower walls. For 129×80 grid cells, $\kappa = 1.066$ was used to obtain the maximum grid aspect ratio $\mathrm{AR}_{\max} = 40$, while $\kappa = 1.00774$ was used for obtaining $\mathrm{AR}_{\max} = 200$.

The first example is for a low Reynolds number (Re = 1600) laminar flow with a mild grid clustering (maximum cell aspect ratio $\mathrm{AR}_{\max} = 10$). For the IRS in this case it was found that $\varepsilon = 2.5$ produced the fastest convergence. The DMR and the SBMR methods were applied once after every 30 iterations, while the LSBMR method was applied once after every 100 iterations. The DMR method combined three consecutive previous iteration results, while the SBMR and LSBMR methods utilized results from two solutions at 10 and 20 iteration steps apart, respectively. The IRS reduced the number of iterations by 44%, while the DMR, LSBMR and SBMR methods reduced the number of iterations by 53%, 65% and 79%, respectively (Fig. 1). For this mild grid clustering, the LSBMR method was not as efficient as the SBMR method, but better than the existing IRS and DMR methods. Computing time overheads for the SBMR and the LSBMR methods were negligible, while those of the IRS and the DMR methods were not. In terms of computing time reduction, 34%, 49%, 65% and 78% of savings were achieved by the IRS, DMR, LSBMR and SBMR methods, respectively. Thus, convergence histories versus computational time consistently mirror the convergence histories versus number of iterations.





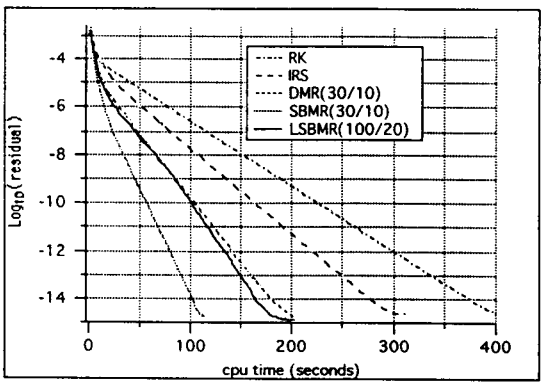


Fig. 1. Convergence histories for a 2-D straight channel flow (L/H = 5, Re = 1600, 60 × 60 grid cells, $AR_{max} = 10$, $\beta = 5$, CFL = 2.8, $\sigma = 0.4$).

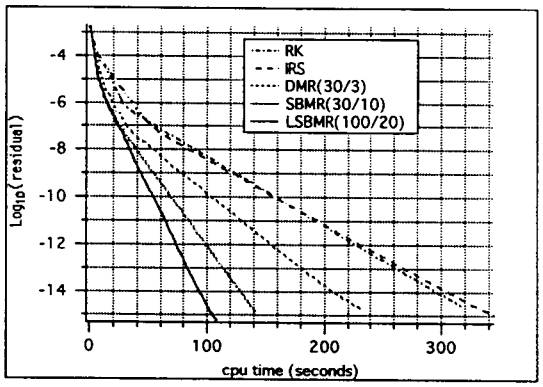


Fig. 2. Convergence histories for a 2-D straight channel flow (L/H = 5, Re = 1600, 60 × 60 grid cells, $AR_{max} = 100$, $\beta = 5$, CFL = 2.8, $\sigma = 0.4$).

As the maximum cell aspect ratio increased to $AR_{max} = 100$, the LSBMR method converged faster than the SBMR method (Fig. 2). This was an expected result, since the LSBMR method allows different optimum α's in the clustered region rather than enforcing fixed acceleration parameters over the whole computational domain. To access the accuracy of our algorithm, the computational results were compared with analytical solutions. Fig. 3(a) shows the relative error in pressure drop at different axial locations. Considering the constant axial pressure gradient, dp/dx, was of the order of 10^{-3} for this Reynolds number, we found that the absolute error in the pressure drop was close to machine zero. In Fig. 3(b), axial velocity error is of the order of 10^{-12} and shows symmetrical distribution about the channel center line which demonstrates that the mass flow is conserved at each cross section. The analytical value of the v-velocity should be zero everywhere and the computed values were of the order of 10^{-13}.

Further increase in maximum cell aspect ratio ($AR_{max} = 10\,000$) slows down the overall convergence. However, the LSBMR method maintains a faster convergence, while other schemes fail to accelerate the basic iteration method (Fig. 4). The history of acceleration coefficients of the LSBMR method demonstrate that the acceleration coefficients in the fine grid region are much larger (Fig. 5(a)) than those in the coarse grid region (Fig. 5(b)). It should also be noticed that all the acceleration coefficients approach the same value as the solution converges to the machine accuracy.

The convergence acceleration schemes were then tested for a high Reynolds number (Re = 1.6 million) turbulent flow in a straight channel. For this high Reynolds number case, we used the Baldwin–Lomax [21] turbulence model and non-reflecting boundary condition was used at the exit of the channel to predict pressure at the exit plane. The inlet flow was assumed to be fully developed, although unlike for the laminar flow cases, the turbulent inlet velocity profile cannot be given analytically. The inlet axial velocity profile was initially assumed to be of the 1/7 power of y/δ and the initial guess for pressure and velocity components throughout the domain were the same as for the

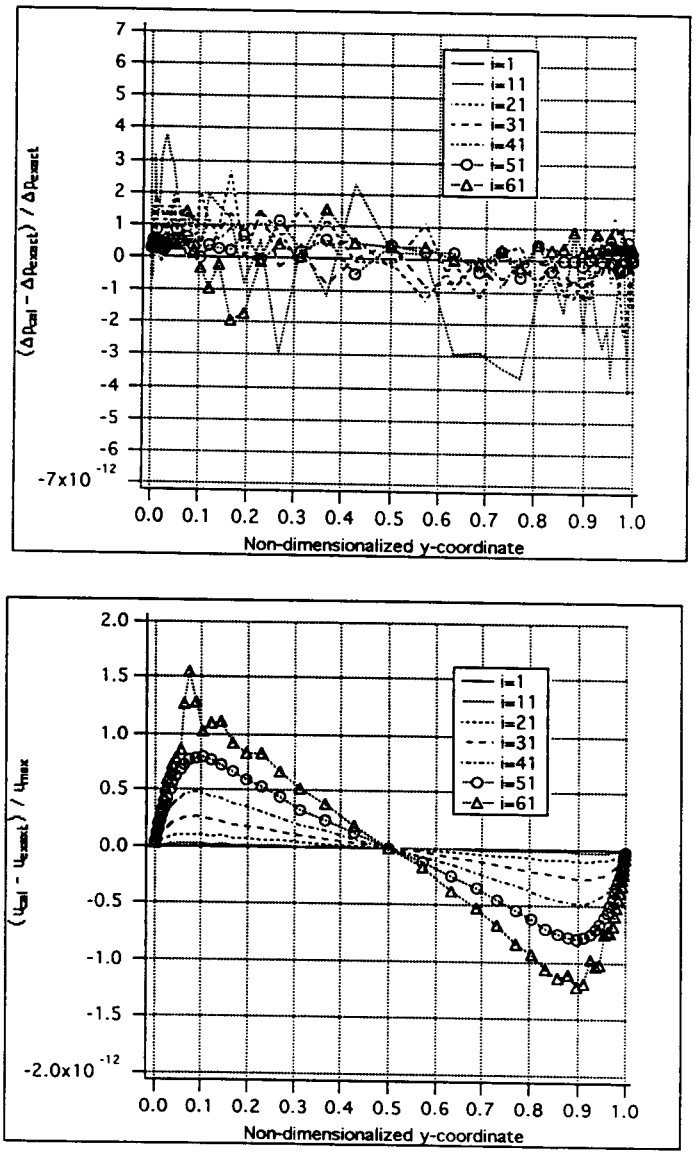


Fig. 3. SBMR method: Relative errors in pressure drop (a) and axial velocity component (b) at different x-locations: $i = 1$ inlet; $i = 61$ exit (Re = 1600, $AR_{max} = 100$, Freq = 30, $n = 10$).

laminar flow examples. We often predicted pressure peaks at the inlet corners when the inlet velocity profile was specified and kept fixed. These pressure peaks were found to be sensitive to the specified inlet velocity profile and their magnitudes increased as the grid became more clustered. Unfortunately, these pressure peaks significantly slowed down the convergence. To circumvent this problem, the inlet velocity profile was slightly modified after each iteration by replacing it with an average of the computed velocity profiles at several immediate locations downstream from the inlet. With this minor modification, the pressure peaks at the inlet corners were quickly eliminated and convergence and robustness of the code significantly improved. With this grid clustering, several grid points could be located within the laminar sublayer. The LSBMR method used the results at 30 iteration levels apart and $\varepsilon = 0.5$ was used for the IRS. Convergence histories for $AR_{max} = 1000$ demonstrate (Fig. 6) that the LSBMR method consistently outperformed other acceleration schemes. When the maximum cell aspect ratio was increased to $AR_{max} = 10\,000$, the LSBMR converged consistently faster than other schemes (Fig. 7).

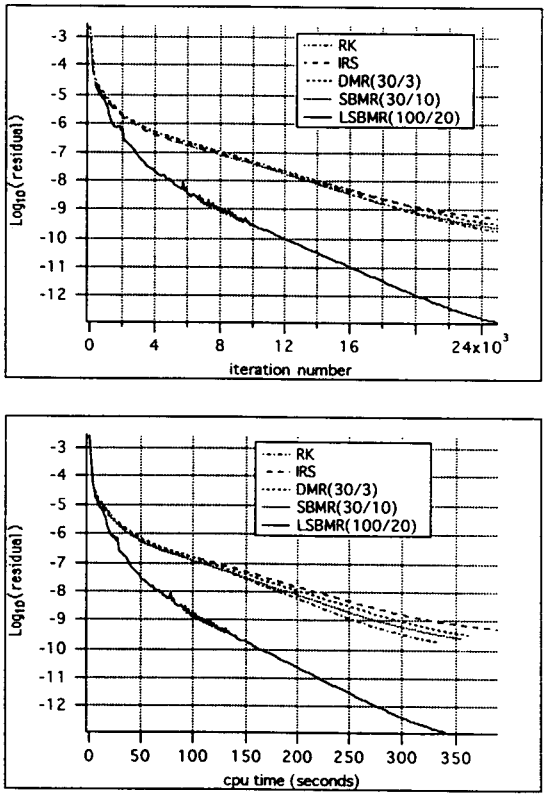


Fig. 4. Convergence histories for a 2-D straight channel flow (L/H = 5, Re = 1600, 60 × 60 grid cells, AR_{max} = 10 000, β = 5, CFL = 2.8, σ = 0.4).

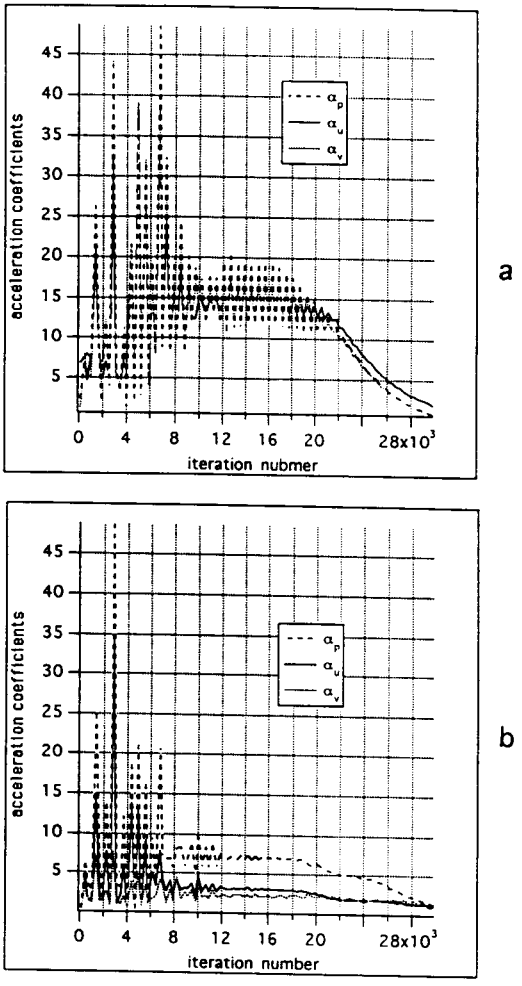


Fig. 5. LSBMR method: Acceleration coefficients for a 2-D straight channel flow at (a) j = 2, and (b) j = 31 (AR_{max} = 10 000, Re = 1600, Freq = 100, n = 20).

As an example of a complex flow with a separated region inside the computational domain, a flow in a U-shaped channel was calculated. The geometry of the channel was discretized with 129 × 80 grid cells that were symmetrically clustered towards the channel walls. In this test case we used Re = 200, CFL = 2.8, von Neumann number = 0.4 and β = 10. Anticipating a recirculating flow pattern at the exit of the U-shaped channel, we used a non-reflecting [14, 15] boundary condition at the exit boundary. A high-pressure region was found at the outer wall of the turning section while a low-pressure region was located at the inner part (Fig. 8(a)). The adverse pressure gradient along the inner part of the turning section resulted in an open-end flow separation downstream of the turning section (Fig. 8(b)) which was successfully predicted because we used the correct form of the non-reflecting exit boundary condition. The LSBMR method used the results at 30 iteration levels apart. The convergence histories for the maximum cell aspect ratio of AR_{max} = 40 for this complex flow example show (Fig. 9) that IRS (with ε = 0.1 for the best performance with AR_{max} = 40) performed only slightly better than the basic RK method. The LSBMR and the DMR methods reduced the number of iterations by 34%, while the DMR reduced it by 20%. When the grid was further clustered toward the channel walls to give maximum cell aspect ratio of AR_{max} = 200, LSBMR method maintained the fastest convergence (Fig. 10), while the DMR and the SBMR methods were not as fast as in the previous lower cell aspect ratio case. Again, the IRS (with ε = 1.0 for the best performance with AR_{max} = 200) did not yield any noticeable acceleration for this test case.

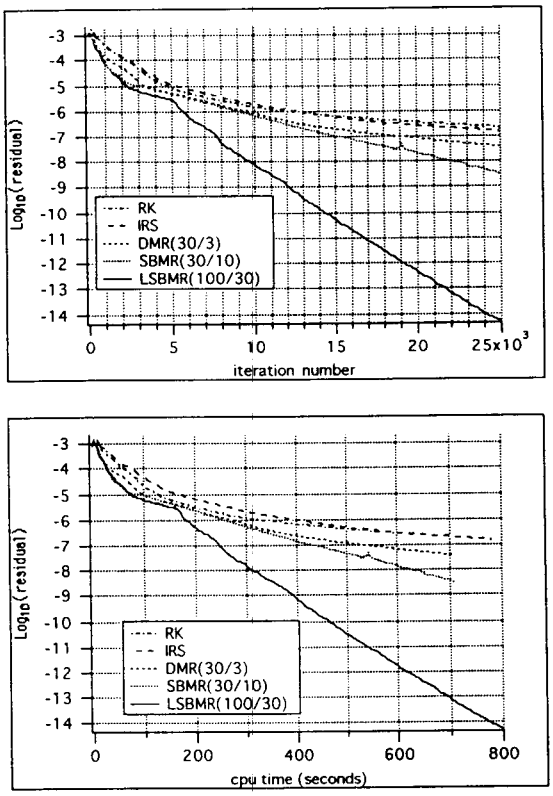


Fig. 6. Convergence histories for a 2-D straight channel flow ($L/H = 10$, $Re = 1.6 \times 10^6$, 60×120 grid cells, $AR_{max} = 1000$, $\beta = 5$, CFL = 2.0, $\sigma = 0.4$).

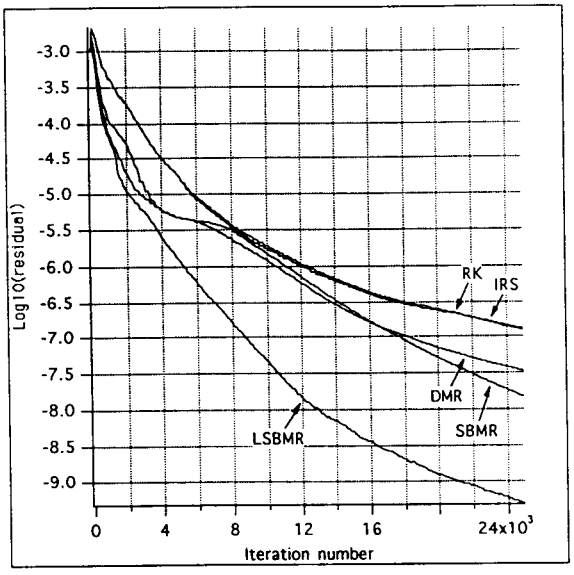


Fig. 7. Convergence histories for a 2-D straight channel flow ($L/H = 10$, $Re = 1.6 \times 10^6$, 60×120 grid cells, $AR_{max} = 10\,000$, $\beta = 5$, CFL = 2.0, $\sigma = 0.4$).

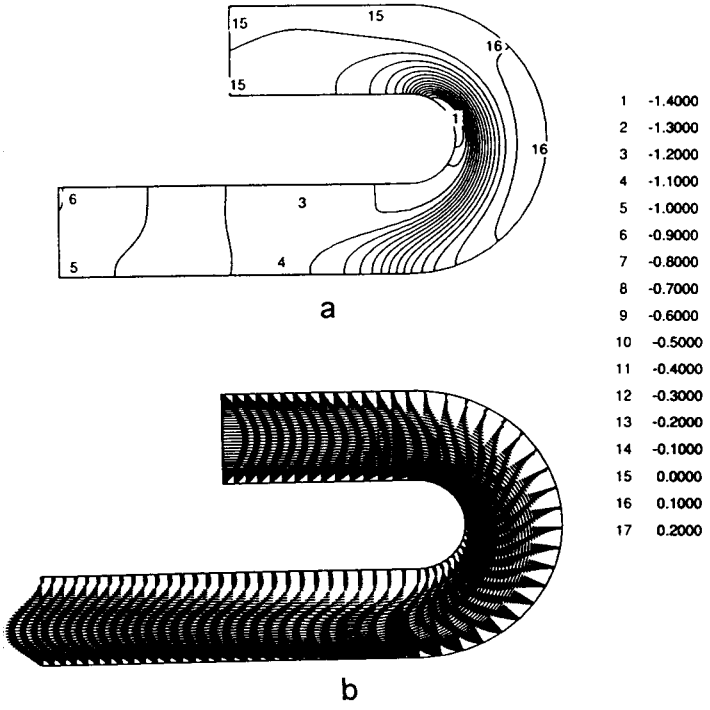
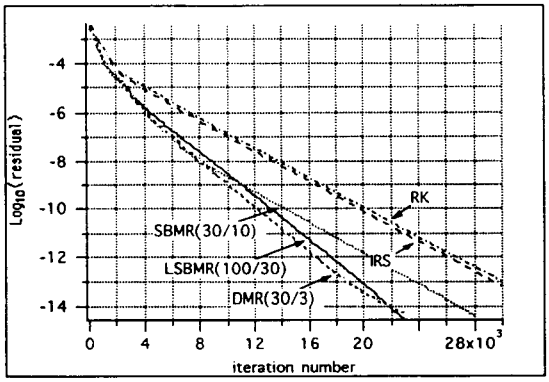
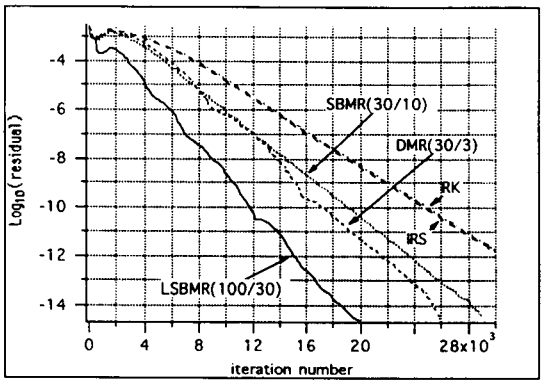


Fig. 8. Isobars (a) and velocity vector field (b) computed for a U-shaped channel flow ($Re = 200$, $AR_{max} = 200$, 129×80 grid cells).





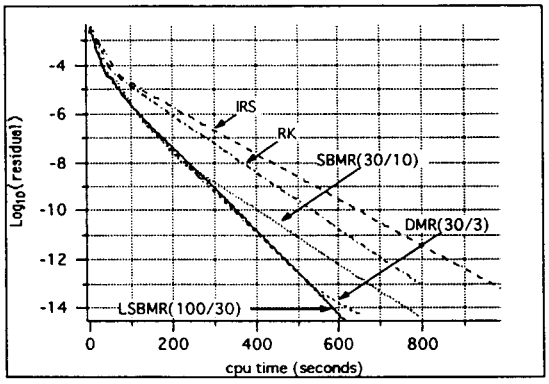


Fig. 9. Convergence histories for a U-shaped channel flow ($Re = 200$, $AR_{max} = 40$, $\beta = 10$, CFL = 2.8, $\sigma = 0.4$, 129×80 grid cells).

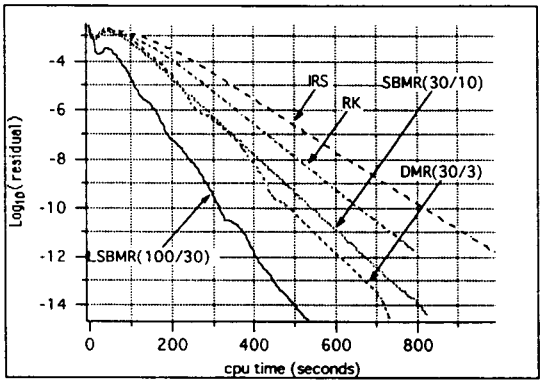


Fig. 10. Convergence histories for a U-shaped channel flow ($Re = 200$, $AR_{max} = 200$, $\beta = 10$, CFL = 2.8, $\sigma = 0.4$, 129×80 grid cells).

8. Conclusions

The Sensitivity-Based Minimum Residual (SBMR) and the Line SBMR (LSBMR) methods were developed and applied to accelerate the convergence of the explicit RK algorithm for incompressible Navier–Stokes equations. The methods are easy to comprehend and to implement in the existing computer codes for iterative integration of systems of partial differential equations. Both new acceleration methods consistently enhanced the convergence rate of the basic Runge–Kutta (RK) method and outperformed the Distributed Minimal Residual (DMR) and the Implicit Residual Smoothing (IRS) methods for both laminar and turbulent flows including separation. The advantage of using the LSBMR method become more evident with increased grid clustering. Both new methods require less computer memory than the DMR method. It can be concluded that the SBMR and the LSBMR method enhance efficiency and robustness of the CFD codes even on highly-clustered grids.

Acknowledgement

This work was supported by the grant NAS 8-38861 from NASA MSFC, Huntsville, AL, monitored by Dr. Paul McConnaughey and by the Penn State NASA Propulsion Engineering Research Center, contract NAGW-1356, Supplement 8. Computing time was provided on the NASA Ames NAS Cray-90 facility by Dr. Louis Povinelli of NASA Lewis Research Center. The final text proofreading was performed by Mr. Thomas J. Martin and Ms. Amy Myers.

References

[1] A. Jameson, W. Schmidt and E. Turkel, Numerical solutions of the Euler equations by finite volume methods using Runge–Kutta time-stepping schemes, AIAA Paper 81–1259, Palo Alto, CA, June, 1981.
[2] A. Jameson and T.J. Baker, Solution of the Euler equations for complex configurations, Proc. AIAA 6th Computational Fluid Dynamics Conference, Danvers, MA (June, 1983) 293–302.
[3] E. Turkel, Review of preconditioning methods for fluid dynamics, ICASE, Report No. 92–47, NASA Langley Research Center, Hampton, VA, September, 1992.
[4] M. Hafez, E. Parlette and M.D. Salas, Convergence acceleration of iterative solutions of Euler equations for transonic flow computations, AIAA Paper 85-1641, Cincinnati, OH, July, 1985.
[5] S. Lee and G.S. Dulikravich, Acceleration of iterative algorithms for Euler equations of gasdynamics, AIAA J. 28(5) (1990) 939–942.
[6] S. Lee and G.S. Dulikravich, Distributed Minimal Residual (DMR) method for acceleration of iterative algorithms, Comput. Methods Appl. Mech. Engrg. 86 (1991) 245–262.
[7] S. Lee and G.S. Dulikravich, Accelerated computation of viscous, steady incompressible flow, ASME Paper 89-GT-45, Gas Turbine and Aeroengine Congress and Exposition, Toronto, Canada, 4–8 June, 1989.
[8] S. Lee and G.S. Dulikravich, Accelerated computation of viscous flow with heat transfer, Numer. Heat Trans. Fundamentals, Part B 19 (1991) 223–241.
[9] C.-Y. Huang and G.S. Dulikravich, Fast iterative algorithms based on optimized explicit time-stepping, Comput. Methods Appl. Mech. Engrg. 63 (1987) 15–36.
[10] S.R. Kennon and G.S. Dulikravich, Optimum acceleration factors for iterative solution of linear and non-linear differential systems, Comput. Methods Appl. Mech. Engrg. 47 (1984) 357–367.
[11] Y. Saad and M. Schultz, Conjugate gradient-like algorithms for solving non-symmetric linear systems, Math. Comput. 44(170) (1985) 417–424.
[12] K.-Y. Choi and G.S. Dulikravich, Reliability enhancement of Navier–Stokes codes through convergence enhancement, NASA Propulsion Engineering Research Center of the Penn State, University Park, PA (8–9 September, 1993) 181–186.
[13] K.-Y. Choi and G.S. Dulikravich, Convergence acceleration of iterative algorithms using a sensitivity-based minimum residual (SBMR) method, 12th Workshop for CFD Applications in Rocket Propulsion, NASA MSFC, Huntsville, AL, 19–21 April, 1994.
[14] K.-Y. Choi and G.S. Dulikravich, Global sensitivity-based minimal residual (GSBMR) method for acceleration of iterative convergence rates, Proc. 14th Internat. Conference on Numerical Fluid Dynamics, Bangalore, India, 11–15 July, 1994.
[15] K.-Y. Choi and G.S. Dulikravich, Convergence rate enhancement of Navier–Stokes codes on clustered grids, 6th Annual Symposium of the Penn State—NASA Propulsion Eng. Research Center, NASA Lewis Research Center, Cleveland, OH, 13–14 September, 1994.
[16] K.-Y. Choi, Sensitivity-based minimum residual methods for convergence acceleration of iterative algorithms, Ph.D. Dissertation, Department of Aerospace Engineering, The Pennsylvania State University, December, 1994.
[17] A.J. Chorin, A numerical method for solving incompressible viscous flow problems, J. Comput. Phys. 2 (1967) 12–26.
[18] K.W. Thomson, Time-dependent boundary conditions for hyperbolic systems, II, J. Comput. Phys. 89 (1990) 439–461.
[19] G.S. Dulikravich, V. Ahuja and S. Lee, Modeling three-dimensional solidification with magnetic fields and reduced gravity, Int. J. Heat Mass Trans. 37(5) (1994) 837–853.
[20] D.A. Anderson, J.C. Tannehill and R.H. Pletcher, Computational Fluid Mechanics and Heat Transfer (McGraw-Hill, New York, 1984) 249–250.
[21] B.S. Baldwin and H. Lomax, Thin layer approximation and algebraic model for separated turbulent flows, AIAA Paper 78–257, Huntsville, AL, 16–18 January, 1978.