Commun. Comput. Phys.
doi: 10.4208/cicp.2009.09.055

Vol. **7**, No. 3, pp. 510-533
March 2010

Implicit Discontinuous Galerkin Method for RANS Simulation Utilizing Pointwise Relaxation Algorithm

Kanako Yasue*, Michiko Furudate, Naofumi Ohnishi and Keisuke Sawada

Department of Aerospace Engineering, Graduate School of Engineering, Tohoku University, Sendai 980-8579, Japan.

Received 20 March 2009; Accepted (in revised version) 19 June 2009

Available online 1 September 2009

Abstract. An efficient implicit procedure for the Discontinuous Galerkin (DG) method is developed utilizing a pointwise relaxation algorithm. In the pointwise relaxation, those contributions from the degrees of freedom in own computational cell are accounted for in the implicit matrix inversion. The resulting scheme is shown to be stable with very large CFL numbers for both the Euler and the Navier-Stokes equations for typical test problems. In order to achieve a faster convergence, efforts are also made to reduce computing time of the present method by utilizing a p-multigrid scheme and also by solving a simplified matrix instead of a fully loaded dense matrix in the implicit matrix inversion. A superior performance of the present implicit DG method on the parallel computer using up to 128 PEs is shown in terms of readily achievable scalability and high parallel efficiency. The RANS simulation of turbulent flowfield over AGARD-B model is carried out to show the convergence property and numerical stability of the present implicit DG method for engineering applications.

AMS subject classifications: 76M10, 65M60, 65D30, 65B99

Key words: Discontinuous Galerkin method, pointwise relaxation implicit scheme, viscous compressible flow.

1 Introduction

Unstructured mesh methods are commonly used in obtaining flowfield over complete aircraft configuration because of their easiness in creating computational mesh for highly complicated geometries. These methods are also known to capture shock waves quite

*Corresponding author. *Email addresses:* hoe@cfd.mech.tohoku.ac.jp (K. Yasue), furu@cfd.mech.tohoku.ac.jp (M. Furudate), ohnishi@rhd.mech.tohoku.ac.jp (N. Ohnishi), sawada@cfd.mech.tohoku.ac.jp (K. Sawada)

sharply using an adaptive grid refinement. In these methods, the finite volume formulation is usually chosen because the conservation laws can be rigorously fulfilled for various cell geometries. However, the spatial accuracy of these methods remains usually at most second order. The cause of this lower spatial accuracy can be attributed to the poorly reconstructed dependent variables in the computational cell. Conventional reconstruction using the cell-averaged variables in nearby cells tends to loose its accuracy for unstructured mesh particularly when cell geometries are highly skewed. In order to capture various features of complicated flowfield in practical problems, truly high order reconstruction method for unstructured mesh should be devised for finite volume methods.

Higher order reconstructions for finite volume method may be achieved for unstructured mesh by using the k-exact formulation [1] or the ENO/WENO schemes [2], though with a substantially increased amount of random memory access due to use of a wider stencil. Recently, the DG finite element method [3,4] has received attentions because of its ability in achieving higher order spatial accuracy rigorously even on unstructured mesh. In this method, instead of referring to nearby cells as in the finite volume methods, reconstruction of the dependent variables is realized with desired accuracy using the degrees of freedom (DOFs) which are introduced in each cell and evolved in time. Therefore, higher order spatial accuracy can be achieved in the DG method with minimal stencil. Indeed, it has been shown that the desired spatial accuracy could be achieved even with various cell geometries using the DG method.

The obvious shortcoming of the DG method is its extremely high computational cost. In the DG method, the dependent variables are expressed as a sum of the DOFs (expansion coefficients) multiplied with the corresponding basis functions. The number of equations to be solved in the DG method is given by a multiple of the number of dependent variables and the number of DOFs introduced in each cell. For example, the number of DOFs is 4 for 3D second order case and 20 for 3D fourth order case. Therefore, one needs to solve 100 equations for the latter case. Furthermore since the Gaussian quadrature formula is used to evaluate integrals, sufficient number of Gaussian quadrature points should be allocated both on the cell boundary and inside of the cell volume to assure numerical accuracy. This increases the number of flux evaluations and results in higher computational cost.

In order to reduce the computing cost of the DG method, it is certainly necessary to develop an implicit scheme to accelerate the convergence, particularly for those steady flow problems. It is also necessary to implement the code on vector/parallel computers. Several such attempts regarding the implicit DG method have already been studied. For example, Bassi and Rebay proposed an implicit DG method utilizing GMRES for the Navier-Stokes equations [5], Rasetarinera and Hussaini developed a matrix-free Krylov approach for the Euler equations [6], Hartmann et al. employed GMRES-Newton algorithm for the Euler and the Navier-Stokes equations [7–9], and Dolejší et al. proposed a semi-implicit DG method for the Euler and the Navier-Stokes equations [10, 11]. In developing implicit schemes, it is very important to have a flexible portability and an easier

parallelization capability. This parallelization capability can be maximized if the implicit scheme is totally pointwise, i.e., no connectivity exists with neighboring cells. Because all the information necessary for reconstructing dependent variables are contained within a cell, the DG method is expected to be suitable for the pointwise relaxation implicit algorithm.

In the present study, we develop a pointwise relaxation implicit DG method of second order spatial accuracy for numerical integration of the Euler and the Navier-Stokes equations. We explore the stability and convergence properties of the present method for typical test problems with very large CFL numbers. In order to reduce computing time of the present method, a dense matrix inversion in each cell is approximated by a simplified matrix inversion where higher order modal components are block diagonalized. Further convergence acceleration is attempted by utilizing the p-multigrid scheme [12, 13]. The parallel performance of the present pointwise relaxation implicit DG method is examined.

2 Numerical method

2.1 Discontinuous Galerkin method

Let us consider the Navier-Stokes equations for 3D flowfield written in the conservation form as

$$\frac{\partial Q}{\partial t}+\frac{\partial E}{\partial x}+\frac{\partial F}{\partial y}+\frac{\partial G}{\partial z}=0, \tag{2.1}$$

where Q, $E=E_c-E_v$, $F=F_c-F_v$ and $G=G_c-G_v$ denote the conservative variable and the flux functions in x, y and z directions, respectively, and are given by

$$Q=\begin{pmatrix}\rho\\ \rho u\\ \rho v\\ \rho w\\ e\end{pmatrix},\quad E_c=\begin{pmatrix}\rho u\\ \rho u^2+p\\ \rho uv\\ \rho uw\\ (e+p)u\end{pmatrix},\quad F_c=\begin{pmatrix}\rho v\\ \rho uv\\ \rho v^2+p\\ \rho vw\\ (e+p)v\end{pmatrix},\quad G_c=\begin{pmatrix}\rho w\\ \rho uw\\ \rho vw\\ \rho w^2+p\\ (e+p)w\end{pmatrix},$$

$$E_v=\begin{pmatrix}0\\ \tau_{xx}\\ \tau_{yx}\\ \tau_{zx}\\ \tau_{xx}u+\tau_{xy}v+\tau_{xz}w+\kappa\frac{\partial T}{\partial x}\end{pmatrix},\quad F_v=\begin{pmatrix}0\\ \tau_{xy}\\ \tau_{yy}\\ \tau_{zy}\\ \tau_{yx}u+\tau_{yy}v+\tau_{yz}w+\kappa\frac{\partial T}{\partial y}\end{pmatrix}, \tag{2.2}$$

$$G_v=\begin{pmatrix}0\\ \tau_{xz}\\ \tau_{yz}\\ \tau_{zz}\\ \tau_{zx}u+\tau_{zy}v+\tau_{zz}w+\kappa\frac{\partial T}{\partial z}\end{pmatrix}.$$

The subscript c represents the convective term and v the viscous term. In the above equations, ρ denotes the fluid density, u, v, w the Cartesian velocity components, e the total energy, T the temperature, and κ the coefficient of heat conduction. The viscous stress components are described as

$$\begin{aligned}
\tau_{xx} &= \frac{2}{3}\mu\left(2\frac{\partial u}{\partial x}-\frac{\partial v}{\partial y}-\frac{\partial w}{\partial z}\right),\\
\tau_{yy} &= \frac{2}{3}\mu\left(-\frac{\partial u}{\partial x}+2\frac{\partial v}{\partial y}-\frac{\partial w}{\partial z}\right),\\
\tau_{zz} &= \frac{2}{3}\mu\left(-\frac{\partial u}{\partial x}-\frac{\partial v}{\partial y}+2\frac{\partial w}{\partial z}\right),\\
\tau_{xy} &= \tau_{yx} = \mu\left(\frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}\right),\\
\tau_{xz} &= \tau_{zx} = \mu\left(\frac{\partial u}{\partial z}+\frac{\partial w}{\partial x}\right),\\
\tau_{yz} &= \tau_{zy} = \mu\left(\frac{\partial v}{\partial z}+\frac{\partial w}{\partial y}\right).
\end{aligned} \tag{2.3}$$

In this study, we introduce a coordinate transformation from a computational cell in (x,y,z) space to a reference cell in (ξ,η,ζ) space in order to introduce orthogonal basis functions in the reference space [14, 15]. Note that the transformation is introduced independently in each computational cell.

The transformed equations are then given by

$$\frac{\partial\overline{Q}}{\partial t}+\frac{\partial\overline{E}}{\partial\xi}+\frac{\partial\overline{F}}{\partial\eta}+\frac{\partial\overline{G}}{\partial\zeta}=0, \tag{2.4}$$

where

$$\begin{aligned}
&\overline{Q}=J^{-1}Q, \quad \overline{E}=J^{-1}(\xi_x E+\xi_y F+\xi_z G),\\
&\overline{F}=J^{-1}(\eta_x E+\eta_y F+\eta_z G), \quad \overline{G}=J^{-1}(\zeta_x E+\zeta_y F+\zeta_z G),
\end{aligned}$$

and J^{-1} is the Jacobian of the transformation. The weak formulation can be found by multiplying Eq. (2.4) with a test function $w(\xi,\eta,\zeta)$, and integrating it over the transformed computational cell $\overline{\Omega}$. The integration by part yields

$$\int_{\overline{\Omega}}\frac{\partial w\overline{Q}}{\partial t}d\overline{\Omega}+\int_{\partial\overline{\Omega}}w(\overline{E}n_\xi+\overline{F}n_\eta+\overline{G}n_\zeta)d\partial\overline{\Omega}-\int_{\overline{\Omega}}\left(\overline{E}\frac{\partial w}{\partial\xi}+\overline{F}\frac{\partial w}{\partial\eta}+\overline{G}\frac{\partial w}{\partial\zeta}\right)d\overline{\Omega}=0, \tag{2.5}$$

where $\partial\overline{\Omega}$ denotes the boundary of the domain $\overline{\Omega}$ and n_ξ, n_η, n_ζ the component of unit outward normal vector in ξ, η and ζ direction of $\partial\overline{\Omega}$, respectively. In the DG method, the

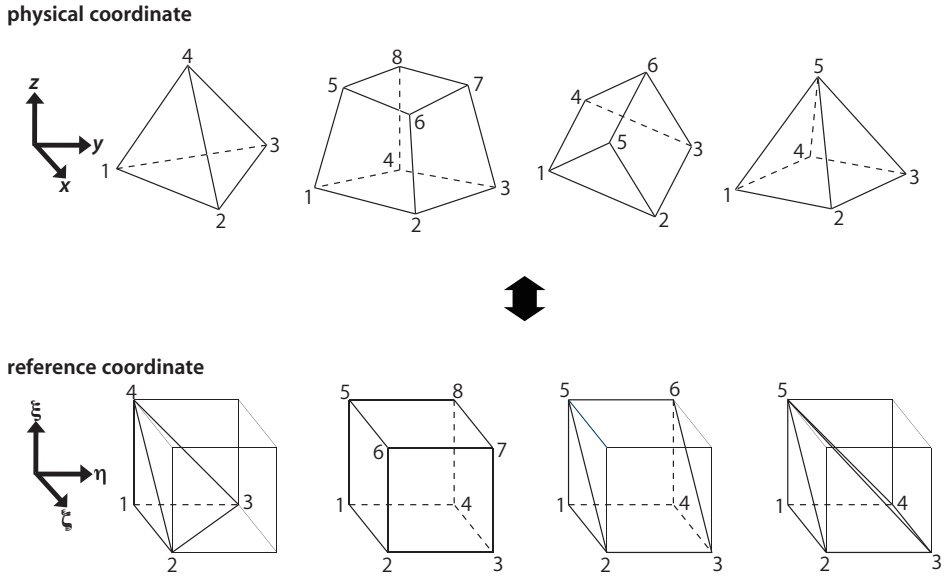


Figure 1: Schematic illustration of 3D mappings from the physical cell in (x, y, z) space to the reference cell in (ξ, η, ζ) space.

transformed conservative variables $\overline{Q}$ is replaced by the approximate solution $\overline{Q}_h$ in the finite element space as

$$\overline{Q}_h(\xi,\eta,\zeta,t) = \sum_j \overline{Q}_j(t)\phi_j(\xi,\eta,\zeta), \tag{2.6}$$

in which $\overline{Q}_j$ denotes the DOFs and ϕ_j the basis functions. The subscript j denotes the number of the DOFs, and varies from 1 to 4 for the 3D second order case. In this study the orthogonal basis function given by Jacobi polynomials are used [14], and described as

$$\begin{aligned}\phi_{lmn}(\xi,\eta,\zeta) = P_l^{0,0}\left(2\frac{1+\xi}{-\eta-\zeta}-1\right)\cdot\left(\frac{-\eta-\zeta}{1-\zeta}\right)^l \\ \cdot P_m^{2l+1,0}\left(2\frac{1+\eta}{1-\zeta}-1\right)\cdot\left(\frac{1-\zeta}{2}\right)^{l+m}\cdot P_n^{2l+2m+2,0}(\zeta),\end{aligned} \tag{2.7}$$

for tetrahedral cells,

$$\phi_{lmn}(\xi,\eta,\zeta) = P_l^{0,0}(\xi)\cdot P_m^{0,0}(\eta)\cdot P_n^{0,0}(\zeta), \tag{2.8}$$

for hexahedral cells,

$$\phi_{lmn}(\xi,\eta,\zeta) = P_l^{0,0}\left(2\frac{1+\xi}{1-\zeta}-1\right)\cdot P_m^{0,0}(\eta)\cdot(1-\zeta)^l\cdot P_n^{2l+1,0}(\zeta), \tag{2.9}$$

for prismatic cells, and

$$\phi_{lmn}(\xi,\eta,\zeta)=P_l^{0,0}\left(2\frac{1+\xi}{1-\zeta}-1\right)\cdot P_m^{0,0}\left(2\frac{1+\eta}{1-\zeta}-1\right)\cdot(1-\zeta)^{l+m}\cdot P_n^{2l+2m+2,0}(\zeta), \tag{2.10}$$

for pyramidal cells, where

$$P_\gamma^{\alpha,\beta}(z)=\frac{(-1)^\gamma}{2^\gamma\gamma!}(1-z)^{-\alpha}(1+z)^{-\beta}\frac{d^\gamma}{dz^\gamma}\Big((1-z)^{\alpha+\gamma}(1+z)^{\beta+\gamma}\Big). \tag{2.11}$$

Subscripts l, m and n specify the order of polynomials. For the 3D second order case, four DOFs are needed for all cell types because the hierarchical modal basis functions are used in this study. The combinations of (l,m,n) are $(0,0,0)$ for the constant term and $(1,0,0)$, $(0,1,0)$ and $(0,0,1)$ for the linear terms. Substituting Eq. (2.6) into Eq. (2.5), and replacing the test function w by the basis function $\phi_i(\xi,\eta,\zeta)$, the weak form of the governing equations is then given by

$$\begin{aligned}\sum_j\frac{d\overline{Q}_j}{dt}I_{ij}+\int_{\partial\overline{\Omega}}\phi_i\left(\overline{E}\left(\overline{Q}_h\right)n_\xi+\overline{F}\left(\overline{Q}_h\right)n_\eta+\overline{G}\left(\overline{Q}_h\right)n_\zeta\right)d\partial\overline{\Omega}\\ -\int_{\overline{\Omega}}\left(\overline{E}\left(\overline{Q}_h\right)\right)\frac{\partial\phi_i}{\partial\xi}+\overline{F}\left(\overline{Q}_h\right)\right)\frac{\partial\phi_i}{\partial\eta}+\overline{G}\left(\overline{Q}_h\right)\right)\frac{\partial\phi_i}{\partial\zeta}\Big)d\overline{\Omega}=0,\end{aligned} \tag{2.12}$$

where

$$I_{ij}=\int_{\overline{\Omega}}\phi_i\phi_j d\overline{\Omega}$$

denotes the mass matrix that becomes diagonal when the basis functions possess orthogonality.

The surface and the volume integrals are evaluated using the Gaussian quadrature formulae. The surface integral in Eq. (2.12) is evaluated in the reference coordinate system (see Fig. 1) in order to ensure that Gaussian quadrature points on both side of interface refer to the identical point in the physical coordinate system. The number of Gaussian quadrature points for the second order case is four for both the triangular and quadrilateral faces. On the other hand, as to the volume integrals, we first transform the cell geometries in the reference coordinate system to a cubic in the transformed coordinate system (not shown in Fig. 1) in unified manner [14]. The volume integral is evaluated for the cubic cell in this transformed space for which we assign eight Gaussian quadrature points for the second order case.

In the DG method, reconstruction of the dependent variables is made independently in each computational cell. This makes the dependent variables discontinuous at the cell interface. The conventional approximate Riemann solver is then applied for the evaluation of the numerical flux function, and the AUSM-DV upwind scheme [16] is chosen in this study. The viscous terms are discretized according to BR2 formulation [17]. Then,

Eq. (2.12) becomes

$$\sum_j \frac{d\overline{Q}_j}{dt} I_{ij} + \int_{\partial\overline{\Omega}} \phi_i \left(\overline{E} n_\xi + \overline{F} n_\eta + \overline{G} n_\zeta\right) d\partial\overline{\Omega} - \int_{\overline{\Omega}} \left(\overline{E}\frac{\partial \phi_i}{\partial \xi} + \overline{F}\frac{\partial \phi_i}{\partial \eta} + \overline{G}\frac{\partial \phi_i}{\partial \zeta}\right) d\overline{\Omega}$$
$$- \int_{\partial\overline{\Omega}} \phi_i \overline{\delta}_n d\partial\overline{\Omega} + \int_{\overline{\Omega}} \left(\overline{\delta}_\xi \frac{\partial \phi_i}{\partial \xi} + \overline{\delta}_\eta \frac{\partial \phi_i}{\partial \eta} + \overline{\delta}_\zeta \frac{\partial \phi_i}{\partial \zeta}\right) d\overline{\Omega} = 0. \tag{2.13}$$

The local lifting terms $\overline{\delta}_\xi$, $\overline{\delta}_\eta$ and $\overline{\delta}_\zeta$ for interface f are weakly defined by

$$\int_{\overline{\Omega}} \left(\psi_\xi \overline{\delta}^f_\xi + \psi_\eta \overline{\delta}^f_\eta + \psi_\zeta \overline{\delta}^f_\zeta\right) d\overline{\Omega} = \int_{\partial\overline{\Omega}_f} \psi_\xi \left(n_\xi \overline{A}_\xi + n_\eta \overline{A}_\eta + n_\zeta \overline{A}_\zeta\right) J^{-1} \frac{Q^R - Q^L}{2} d\partial\overline{\Omega}$$
$$+ \int_{\partial\overline{\Omega}_f} \psi_\eta \left(n_\xi \overline{B}_\xi + n_\eta \overline{B}_\eta + n_\zeta \overline{B}_\zeta\right) J^{-1} \frac{Q^R - Q^L}{2} d\partial\overline{\Omega}$$
$$+ \int_{\partial\overline{\Omega}_f} \psi_\zeta \left(n_\xi \overline{C}_\xi + n_\eta \overline{C}_\eta + n_\zeta \overline{C}_\zeta\right) J^{-1} \frac{Q^R - Q^L}{2} d\partial\overline{\Omega}, \tag{2.14}$$

where ψ_ξ, ψ_η and ψ_ζ are test functions. Here we assume that the viscous flux functions $\overline{E}_v$, $\overline{F}_v$ and $\overline{G}_v$ can be written respectively as

$$\begin{aligned}
\overline{E}_v &= J^{-1}\left(\xi_x E_v + \xi_y F_v + \xi_z G_v\right) = \overline{A}_\xi \frac{\partial Q}{\partial \xi} + \overline{A}_\eta \frac{\partial Q}{\partial \eta} + \overline{A}_\zeta \frac{\partial Q}{\partial \zeta}, \\
\overline{F}_v &= J^{-1}\left(\eta_x E_v + \eta_y F_v + \eta_z G_v\right) = \overline{B}_\xi \frac{\partial Q}{\partial \xi} + \overline{B}_\eta \frac{\partial Q}{\partial \eta} + \overline{B}_\zeta \frac{\partial Q}{\partial \zeta}, \\
\overline{G}_v &= J^{-1}\left(\zeta_x E_v + \zeta_y F_v + \zeta_z G_v\right) = \overline{C}_\xi \frac{\partial Q}{\partial \xi} + \overline{C}_\eta \frac{\partial Q}{\partial \eta} + \overline{C}_\zeta \frac{\partial Q}{\partial \zeta}.
\end{aligned} \tag{2.15}$$

The trace values of the dependent variable at the cell interface are given by Q^L and Q^R in Eq. (2.14) where the superscript L denotes the inside of the cell and R the outside. The fourth term in Eq. (2.13) involving the projected local lifting term $\overline{\delta}_n$ is determined weakly for the interface f when

$$\psi_\xi = n_\xi \phi_i, \quad \psi_\eta = n_\eta \phi_i, \quad \psi_\zeta = n_\zeta \phi_i$$

are substituted in Eq. (2.14), while the last term in Eq. (2.13) can be found when

$$\psi_\xi = \frac{\partial \phi_i}{\partial \xi}, \quad \psi_\eta = \frac{\partial \phi_i}{\partial \eta}, \quad \psi_\zeta = \frac{\partial \phi_i}{\partial \zeta}$$

are substituted in Eq. (2.14).

For calculation of turbulent flowfield, one equation turbulence model proposed by Spalart and Allmaras is employed [18]. The transport equation of the Spalart-Allmaras turbulence model is also transformed into reference space, and the working variable of the model is expanded similarly in the finite element space.

2.2 Pointwise relaxation implicit scheme

We first consider the linearization of the convective flux function given by

$$\begin{aligned}
\overline{E}_c^{n+1} &\cong \overline{E}_c^n + \frac{\partial \overline{E}_c^n}{\partial \overline{Q}} \Delta \overline{Q} = \overline{E}_c^n + \left(\xi_x A_c + \xi_y B_c + \xi_z C_c\right) \Delta \overline{Q}, \\
\overline{F}_c^{n+1} &\cong \overline{F}_c^n + \frac{\partial \overline{F}_c^n}{\partial \overline{Q}} \Delta \overline{Q} = \overline{F}_c^n + \left(\eta_x A_c + \eta_y B_c + \eta_z C_c\right) \Delta \overline{Q}, \\
\overline{G}_c^{n+1} &\cong \overline{G}_c^n + \frac{\partial \overline{G}_c^n}{\partial \overline{Q}} \Delta \overline{Q} = \overline{G}_c^n + \left(\zeta_x A_c + \zeta_y B_c + \zeta_z C_c\right) \Delta \overline{Q},
\end{aligned} \tag{2.16}$$

where $A_c = \partial E_c / \partial Q$, $B_c = \partial F_c / \partial Q$ and $C_c = \partial G_c / \partial Q$ are the Jacobian matrices of the convective flux function in (x,y,z) space, and n denotes the time step. We then approximate the flux integral of the convective terms in Eq. (2.13) as

$$\begin{aligned}
&\int_{\partial \overline{\Omega}} \phi_i \left(\overline{E}_c^{n+1} n_\xi + \overline{F}_c^{n+1} n_\eta + \overline{G}_c^{n+1} n_\zeta \right) d\partial\overline{\Omega} \\
&\cong \int_{\partial \overline{\Omega}} \phi_i \left(\overline{E}_c^{n} n_\xi + \overline{F}_c^{n} n_\eta + \overline{G}_c^{n} n_\zeta \right) d\partial\overline{\Omega} + \int_{\partial \overline{\Omega}} \phi_i \left(\sigma_x A_c + \sigma_y B_c + \sigma_z C_c \right)^+ \Delta \overline{Q} d\partial\overline{\Omega},
\end{aligned} \tag{2.17}$$

where

$$\sigma_x = \xi_x n_\xi + \eta_x n_\eta + \zeta_x n_\zeta, \quad \sigma_y = \xi_y n_\xi + \eta_y n_\eta + \zeta_y n_\zeta, \quad \sigma_z = \xi_z n_\xi + \eta_z n_\eta + \zeta_z n_\zeta.$$

The positive projection of the Jacobian matrix D of the convective flux function is given by

$$D^+ = \kappa (D + \lambda_{\max} I)/2,$$

where

$$D = \sigma_x A_c + \sigma_y B_c + \sigma_z C_c$$

and $\lambda_{\max}$ is the maximum eigenvalue given by

$$\lambda_{\max} = (|u\sigma_x + v\sigma_y + w\sigma_z| + c\sqrt{\sigma_x^2 + \sigma_y^2 + \sigma_z^2}), \tag{2.18}$$

in which c is the sound velocity. Similar to that in LU-SGS scheme [19], we choose κ=1.05. Note that the temporal variation of the transformed DOFs is related to $\Delta\overline{Q}$ by

$$\Delta \overline{Q} = \sum_j \Delta \overline{Q}_j \phi_j. \tag{2.19}$$

Substituting Eq. (2.19) into Eq. (2.17), we obtain

$$\begin{aligned}
&\int_{\partial \overline{\Omega}} \phi_i \left(\overline{E}_c^{n+1} n_\xi + \overline{F}_c^{n+1} n_\eta + \overline{G}_c^{n+1} n_\zeta \right) d\partial\overline{\Omega} \\
&\cong \int_{\partial \overline{\Omega}} \phi_i \left(\overline{E}_c^{n} n_\xi + \overline{F}_c^{n} n_\eta + \overline{G}_c^{n} n_\zeta \right) d\partial\overline{\Omega} + \sum_j \int_{\partial \overline{\Omega}} \phi_i \left(\sigma_x A_c + \sigma_y B_c + \sigma_z C_c \right)^+ \Delta \overline{Q}_j \phi_j d\partial\overline{\Omega}.
\end{aligned} \tag{2.20}$$

The linearization of the volume integral in Eq. (2.13) is given as

$$\begin{aligned}
&\int_{\overline{\Omega}}\left(\overline{E}_c^{n+1}\frac{\partial\phi_i}{\partial\xi}+\overline{F}_c^{n+1}\frac{\partial\phi_i}{\partial\eta}+\overline{G}_c^{n+1}\frac{\partial\phi_i}{\partial\zeta}\right)d\overline{\Omega}\\
&\cong\int_{\overline{\Omega}}\left(\overline{E}_c^{n}\frac{\partial\phi_i}{\partial\xi}+\overline{F}_c^{n}\frac{\partial\phi_i}{\partial\eta}+\overline{G}_c^{n}\frac{\partial\phi_i}{\partial\zeta}\right)d\overline{\Omega}+\int_{\overline{\Omega}}\left(\omega_x^iA_c+\omega_y^iB_c+\omega_z^iC_c\right)\Delta\overline{Q}d\overline{\Omega}\\
&=\int_{\overline{\Omega}}\left(\overline{E}_c^{n}\frac{\partial\phi_i}{\partial\xi}+\overline{F}_c^{n}\frac{\partial\phi_i}{\partial\eta}+\overline{G}_c^{n}\frac{\partial\phi_i}{\partial\zeta}\right)d\overline{\Omega}+\sum_j\int_{\overline{\Omega}}\left(\omega_x^iA_c+\omega_y^iB_c+\omega_z^iC_c\right)\Delta\overline{Q}_j\phi_jd\overline{\Omega},
\end{aligned}\tag{2.21}$$

where

$$\begin{aligned}
\omega_x^i&=\xi_x\frac{\partial\phi_i}{\partial\xi}+\eta_x\frac{\partial\phi_i}{\partial\eta}+\zeta_x\frac{\partial\phi_i}{\partial\zeta},\\
\omega_y^i&=\xi_y\frac{\partial\phi_i}{\partial\xi}+\eta_y\frac{\partial\phi_i}{\partial\eta}+\zeta_y\frac{\partial\phi_i}{\partial\zeta},\\
\omega_z^i&=\xi_z\frac{\partial\phi_i}{\partial\xi}+\eta_z\frac{\partial\phi_i}{\partial\eta}+\zeta_z\frac{\partial\phi_i}{\partial\zeta}.
\end{aligned}\tag{2.22}$$

Next, the linearization of the viscous terms and also the lifting terms from BR2 formulation are described. For the viscous terms, almost the same procedures used for the convective terms are employed. The linearizations of the flux integral and the volume integral in Eq. (2.13) for the viscous terms are approximated as

$$\begin{aligned}
&\int_{\partial\overline{\Omega}}\phi_i\left(\overline{E}_v^{n+1}n_\xi+\overline{F}_v^{n+1}n_\eta+\overline{G}_v^{n+1}n_\zeta\right)d\partial\overline{\Omega}\\
&\cong\int_{\partial\overline{\Omega}}\phi_i\left(\overline{E}_v^{n}n_\xi+\overline{F}_v^{n}n_\eta+\overline{G}_v^{n}n_\zeta\right)d\partial\overline{\Omega}+\sum_j\int_{\partial\overline{\Omega}}\phi_i\left(\sigma_xA_v+\sigma_yB_v+\sigma_zC_v\right)\Delta\overline{Q}_j\phi_jd\partial\overline{\Omega},
\end{aligned}\tag{2.23}$$

and

$$\begin{aligned}
&\int_{\overline{\Omega}}\left(\overline{E}_v^{n+1}\frac{\partial\phi_i}{\partial\xi}+\overline{F}_v^{n+1}\frac{\partial\phi_i}{\partial\eta}+\overline{G}_v^{n+1}\frac{\partial\phi_i}{\partial\zeta}\right)d\overline{\Omega}\\
&\cong\int_{\overline{\Omega}}\left(\overline{E}_v^{n}\frac{\partial\phi_i}{\partial\xi}+\overline{F}_v^{n}\frac{\partial\phi_i}{\partial\eta}+\overline{G}_v^{n}\frac{\partial\phi_i}{\partial\zeta}\right)d\overline{\Omega}+\sum_j\int_{\overline{\Omega}}\left(\omega_x^iA_v+\omega_y^iB_v+\omega_z^iC_v\right)\Delta\overline{Q}_j\phi_jd\overline{\Omega},
\end{aligned}\tag{2.24}$$

respectively, where

$$A_v=\partial E_v/\partial Q,\quad B_v=\partial F_v/\partial Q,\quad C_v=\partial G_v/\partial Q.$$

In this study, we further assume that the local lifting term can be expanded using the basis functions as

$$\overline{\delta}_n=\sum_j\phi_j\overline{\delta}_{n_j}.\tag{2.25}$$

This approximates the integration of the local lifting term as

$$\begin{aligned}
&\int_{\partial\overline{\Omega}}\phi_i\overline{\delta}_n^{n+1}d\partial\overline{\Omega}\\
&\cong\int_{\partial\overline{\Omega}}\phi_i\overline{\delta}_n^{n}d\partial\overline{\Omega}+\int_{\partial\overline{\Omega}}\phi_i\frac{\partial}{\partial\overline{Q}}\left(\sum_k\phi_k\overline{\delta}_{n_k}\right)\Delta\overline{Q}d\partial\overline{\Omega}\\
&=\int_{\partial\overline{\Omega}}\phi_i\overline{\delta}_n^{n}d\partial\overline{\Omega}+\int_{\partial\overline{\Omega}}\phi_i\left(\sum_k\phi_k\frac{\partial\overline{\delta}_{n_k}}{\partial\overline{Q}}\right)\sum_j\phi_j\Delta\overline{Q}_jd\partial\overline{\Omega},
\end{aligned}\tag{2.26}$$

in which the derivative of the expansion coefficient is given approximately as

$$\begin{aligned}
\frac{\partial\overline{\delta}_{n_i}}{\partial\overline{Q}}=\frac{1}{I_{ii}}\Bigg\{&\int_{\partial\overline{\Omega}}\phi_in_\xi\left(n_\xi\overline{A}_\xi+n_\eta\overline{A}_\eta+n_\zeta\overline{A}_\zeta\right)\left(-\frac{1}{2}\right)d\partial\overline{\Omega}\\
&+\int_{\partial\overline{\Omega}}\phi_in_\eta\left(n_\xi\overline{B}_\xi+n_\eta\overline{B}_\eta+n_\zeta\overline{B}_\zeta\right)\left(-\frac{1}{2}\right)d\partial\overline{\Omega}\\
&+\int_{\partial\overline{\Omega}}\phi_in_\zeta\left(n_\xi\overline{C}_\xi+n_\eta\overline{C}_\eta+n_\zeta\overline{C}_\zeta\right)\left(-\frac{1}{2}\right)d\partial\overline{\Omega}\Bigg\},
\end{aligned}\tag{2.27}$$

where we freeze the coefficient matrices.

The volume integral for the local lifting term is then approximated as

$$\begin{aligned}
&\int_{\overline{\Omega}}\left(\overline{\delta}_\xi\frac{\partial\phi_i}{\partial\xi}+\overline{\delta}_\eta\frac{\partial\phi_i}{\partial\eta}+\overline{\delta}_\zeta\frac{\partial\phi_i}{\partial\zeta}\right)^{n+1}d\overline{\Omega}\\
&=\int_{\partial\overline{\Omega}}\left(A^*J^{-1}\frac{Q^R-Q^L}{2}\right)^{n+1}d\partial\overline{\Omega}\\
&\cong\int_{\partial\overline{\Omega}}\left(A^*J^{-1}\frac{Q^R-Q^L}{2}\right)^{n}d\partial\overline{\Omega}+\int_{\partial\overline{\Omega}}A^*\left(-\frac{1}{2}\right)\Delta\overline{Q}d\partial\overline{\Omega}\\
&=\int_{\partial\overline{\Omega}}\left(A^*J^{-1}\frac{Q^R-Q^L}{2}\right)^{n}d\partial\overline{\Omega}+\int_{\partial\overline{\Omega}}A^*\left(-\frac{1}{2}\right)\sum_j\phi_j\Delta\overline{Q}_jd\partial\overline{\Omega},
\end{aligned}\tag{2.28}$$

where

$$\begin{aligned}
A^*=&\frac{\partial\phi_i}{\partial\xi}\left(\overline{A}_\xi n_\xi+\overline{A}_\eta n_\eta+\overline{A}_\zeta n_\zeta\right)+\frac{\partial\phi_i}{\partial\eta}\left(\overline{B}_\xi n_\xi+\overline{B}_\eta n_\eta+\overline{B}_\zeta n_\zeta\right)\\
&+\frac{\partial\phi_i}{\partial\zeta}\left(\overline{C}_\xi n_\xi+\overline{C}_\eta n_\eta+\overline{C}_\zeta n_\zeta\right).
\end{aligned}\tag{2.29}$$

Finally, the time derivative term in Eq. (2.13) is approximated to first order in time as

$$\sum_j\frac{d\overline{Q}_j}{dt}I_{ij}=\frac{1}{\Delta t}\sum_jI_{ij}\Delta\overline{Q}_j.\tag{2.30}$$

We now obtain the algebraic equation for the temporal change of the transformed DOFs $(\Delta\overline{Q}_i,\ i=1,2,\cdots)$ in each computational cell as

$$\sum_j \boldsymbol{\mathcal{M}}_{ij}\Delta\overline{Q}_j=\boldsymbol{\mathcal{R}}_i, \tag{2.31}$$

where

$$\begin{aligned}\boldsymbol{\mathcal{M}}_{ij}=&\frac{1}{\Delta t}I_{ij}+\int_{\partial\overline{\Omega}}\phi_i(\sigma_x A_c+\sigma_y B_c+\sigma_z C_c)^+\phi_j d\partial\overline{\Omega}-\int_{\overline{\Omega}}\left(\omega_x^i A_c+\omega_y^i B_c+\omega_z^i C_c\right)\phi_j d\overline{\Omega}\\&-\int_{\partial\overline{\Omega}}\phi_i(\sigma_x A_v+\sigma_y B_v+\sigma_z C_v)\phi_j d\partial\overline{\Omega}+\int_{\overline{\Omega}}\left(\omega_x^i A_v+\omega_y^i B_v+\omega_z^i C_v\right)\phi_j d\overline{\Omega}\\&-\int_{\partial\overline{\Omega}}\phi_i\left(\sum_k\phi_k\frac{\partial\overline{\delta}_{n_k}}{\partial\overline{Q}}\right)\phi_j d\partial\overline{\Omega}+\int_{\partial\overline{\Omega}}A^*\left(-\frac{1}{2}\right)\phi_j d\partial\overline{\Omega},\end{aligned} \tag{2.32}$$

and

$$\begin{aligned}\boldsymbol{\mathcal{R}}_i=&-\int_{\partial\overline{\Omega}}\phi_i\left(\overline{E}^n n_\xi+\overline{F}^n n_\eta+\overline{G}^n n_\zeta\right)d\partial\overline{\Omega}+\int_{\overline{\Omega}}\left(\overline{E}^n\frac{\partial\phi_i}{\partial\xi}+\overline{F}^n\frac{\partial\phi_i}{\partial\eta}+\overline{G}^n\frac{\partial\phi_i}{\partial\zeta}\right)d\overline{\Omega}\\&+\int_{\partial\overline{\Omega}}\phi_i\overline{\delta}_n^n d\partial\overline{\Omega}-\int_{\partial\overline{\Omega}}\left(A^*J^{-1}\frac{Q^R-Q^L}{2}\right)^n\phi_j d\partial\overline{\Omega}.\end{aligned} \tag{2.33}$$

The matrix $\boldsymbol{\mathcal{M}}$ to be inverted in each computational cell has 20×20 components for 3D problems for a second order case.

We note that the present pointwise relaxation implicit DG method is really an explicit-like scheme in the sense that Eq. (2.31) is solved in each computational cell independently. Therefore it is particularly interesting to observe how the solution evolves in time when a very large CFL number is assumed. This is examined by applying the present scheme to a 3D linear advection problem. The computed results will be shown in the later section. The obtained convergence histories shown therein indicate that the residual slowly decreases for a certain number of iterations at the beginning of the calculation. In that period, the incident wave from the inflow boundary propagates across the computational domain one cell by another per time step. After that, the residual begins to decrease quickly to a machine zero. The former period may be termed as the time evolving phase, and the latter as the relaxation phase. Any implicit scheme exhibits more or less the same tendency, but it seems that the time evolving phase is more evident for the present pointwise relaxation implicit scheme. It should be noted that a conventional implicit scheme probably can achieve a faster convergence than the present one because the information propagates over the entire domain for every time step through implicit matrix inversion. However, we prefer the present pointwise approach for its easier parallelization property even though a larger iteration number is to be needed for convergence.

2.3 Reduction of computational cost

2.3.1 Matrix simplification

A considerable part of the computational cost of the present implicit solver comes from those for setting the matrix components in Eq. (2.32) and inversion of the algebraic equation in Eq. (2.31). By simplifying the dense matrix $\boldsymbol{\mathcal{M}}_{ij}$, one can reduce the computational time and memory size, simultaneously.

The algebraic equation in Eq. (2.31) for the 3D second order case can be written as

$$\begin{pmatrix} \boldsymbol{M}_{11} & \boldsymbol{M}_{12} & \boldsymbol{M}_{13} & \boldsymbol{M}_{14} \\ \boldsymbol{M}_{21} & \boldsymbol{M}_{22} & \boldsymbol{M}_{23} & \boldsymbol{M}_{24} \\ \boldsymbol{M}_{31} & \boldsymbol{M}_{32} & \boldsymbol{M}_{33} & \boldsymbol{M}_{34} \\ \boldsymbol{M}_{41} & \boldsymbol{M}_{42} & \boldsymbol{M}_{43} & \boldsymbol{M}_{44} \end{pmatrix} \begin{pmatrix} \Delta\overline{Q}_1 \\ \Delta\overline{Q}_2 \\ \Delta\overline{Q}_3 \\ \Delta\overline{Q}_4 \end{pmatrix} = \begin{pmatrix} \boldsymbol{R}_1 \\ \boldsymbol{R}_2 \\ \boldsymbol{R}_3 \\ \boldsymbol{R}_4 \end{pmatrix}, \tag{2.34}$$

where $\Delta\overline{Q}_1$ corresponds to the lowest order (constant) term while $\Delta\overline{Q}_2$, $\Delta\overline{Q}_3$ and $\Delta\overline{Q}_4$ are for the first order (linear) terms. The size of each matrix component $\boldsymbol{M}_{ij}$ is 5×5. In the original pointwise relaxation implicit scheme, a fully loaded dense matrix is solved by a direct inversion using LU decomposition. We approximate this algebraic equation to a simpler form as

$$\begin{pmatrix} \boldsymbol{M}_{11} & \boldsymbol{M}_{12} & \boldsymbol{M}_{13} & \boldsymbol{M}_{14} \\ \boldsymbol{M}_{21} & \boldsymbol{M}_{22} & 0 & 0 \\ \boldsymbol{M}_{31} & 0 & \boldsymbol{M}_{33} & 0 \\ \boldsymbol{M}_{41} & 0 & 0 & \boldsymbol{M}_{44} \end{pmatrix} \begin{pmatrix} \Delta\overline{Q}_1 \\ \Delta\overline{Q}_2 \\ \Delta\overline{Q}_3 \\ \Delta\overline{Q}_4 \end{pmatrix} = \begin{pmatrix} \boldsymbol{R}_1 \\ \boldsymbol{R}_2 \\ \boldsymbol{R}_3 \\ \boldsymbol{R}_4 \end{pmatrix}. \tag{2.35}$$

This simplified form is chosen due to an observation that coupling among three first order (linear) terms is not so tight and hence diagonalized form could be applicable to these terms while the coupling between the lower order (constant) term and those linear terms should be retained. Numerical experiments have revealed that above simplified form maintains numerical stability and convergence property of the original one not only for solving the Euler equations but also for the Navier-Stokes equations with one-equation turbulence model. Eq. (2.35) is equivalent to solve

$$\begin{pmatrix} \widehat{\boldsymbol{M}}_{11} & 0 & 0 & 0 \\ \boldsymbol{M}_{21} & \boldsymbol{M}_{22} & 0 & 0 \\ \boldsymbol{M}_{31} & 0 & \boldsymbol{M}_{33} & 0 \\ \boldsymbol{M}_{41} & 0 & 0 & \boldsymbol{M}_{44} \end{pmatrix} \begin{pmatrix} \Delta\overline{Q}_1 \\ \Delta\overline{Q}_2 \\ \Delta\overline{Q}_3 \\ \Delta\overline{Q}_4 \end{pmatrix} = \begin{pmatrix} \widehat{\boldsymbol{R}}_1 \\ \boldsymbol{R}_2 \\ \boldsymbol{R}_3 \\ \boldsymbol{R}_4 \end{pmatrix}, \tag{2.36}$$

where

$$\begin{aligned} \widehat{\boldsymbol{M}}_{11} &= \boldsymbol{M}_{11} - \boldsymbol{M}_{12}\boldsymbol{M}_{22}^{-1}\boldsymbol{M}_{21} - \boldsymbol{M}_{13}\boldsymbol{M}_{33}^{-1}\boldsymbol{M}_{31} - \boldsymbol{M}_{14}\boldsymbol{M}_{44}^{-1}\boldsymbol{M}_{41}, \\ \widehat{\boldsymbol{R}}_1 &= \boldsymbol{R}_1 - \boldsymbol{M}_{12}\boldsymbol{M}_{22}^{-1}\boldsymbol{R}_2 - \boldsymbol{M}_{13}\boldsymbol{M}_{33}^{-1}\boldsymbol{R}_3 - \boldsymbol{M}_{14}\boldsymbol{M}_{44}^{-1}\boldsymbol{R}_4. \end{aligned} \tag{2.37}$$

Therefore, instead of inverting a 20×20 matrix for once, we consider inversion of 5×5 matrix for four times as

$$\begin{aligned}
\Delta\overline{Q}_1 &= \widehat{\boldsymbol{M}}_{11}^{-1}\widehat{\boldsymbol{R}}_1,\\
\Delta\overline{Q}_2 &= \boldsymbol{M}_{22}^{-1}\left(\boldsymbol{R}_2-\boldsymbol{M}_{21}\Delta\overline{Q}_1\right),\\
\Delta\overline{Q}_3 &= \boldsymbol{M}_{33}^{-1}\left(\boldsymbol{R}_3-\boldsymbol{M}_{31}\Delta\overline{Q}_1\right),\\
\Delta\overline{Q}_4 &= \boldsymbol{M}_{44}^{-1}\left(\boldsymbol{R}_4-\boldsymbol{M}_{41}\Delta\overline{Q}_1\right).
\end{aligned} \tag{2.38}$$

We note that a fully block diagonalized case such as to solve

$$\begin{pmatrix} \boldsymbol{M}_{11} & 0 & 0 & 0\\ 0 & \boldsymbol{M}_{22} & 0 & 0\\ 0 & 0 & \boldsymbol{M}_{33} & 0\\ 0 & 0 & 0 & \boldsymbol{M}_{44}\end{pmatrix}\begin{pmatrix}\Delta\overline{Q}_1\\ \Delta\overline{Q}_2\\ \Delta\overline{Q}_3\\ \Delta\overline{Q}_4\end{pmatrix}=\begin{pmatrix}\boldsymbol{R}_1\\ \boldsymbol{R}_2\\ \boldsymbol{R}_3\\ \boldsymbol{R}_4\end{pmatrix}, \tag{2.39}$$

resulted in numerical instability. Furthermore, the block matrix inversion similar to Eq. (2.36) but to solve

$$\begin{pmatrix} \boldsymbol{M}_{11} & 0 & 0 & 0\\ \boldsymbol{M}_{21} & \boldsymbol{M}_{22} & 0 & 0\\ \boldsymbol{M}_{31} & 0 & \boldsymbol{M}_{33} & 0\\ \boldsymbol{M}_{41} & 0 & 0 & \boldsymbol{M}_{44}\end{pmatrix}\begin{pmatrix}\Delta\overline{Q}_1\\ \Delta\overline{Q}_2\\ \Delta\overline{Q}_3\\ \Delta\overline{Q}_4\end{pmatrix}=\begin{pmatrix}\boldsymbol{R}_1\\ \boldsymbol{R}_2\\ \boldsymbol{R}_3\\ \boldsymbol{R}_4\end{pmatrix}, \tag{2.40}$$

exhibited weak instability for the Euler equations where we attempted to obtain a uniform flow using the mesh system prepared for the computation of the laminar boundary layer flow over a flat plate (Fig. 11 appearing in later section), although the slippery wall condition is applied at the flat plate. In the Navier-Stokes simulation using the same mesh, Eq. (2.40) gives a stable numerical procedure even with very large CFL numbers, and can reach a machine zero convergence. A detailed stability analysis of the pointwise relaxation implicit DG method as well as for the higher order cases will appear elsewhere.

2.3.2 *p*-multigrid scheme

As an effective convergence acceleration technique, *h*-multigrid scheme is very popular. In the *h*-multigrid scheme, error component of longer wavelength on a finer grid is selectively decayed using solutions on a coarser grid. In the *p*-multigrid, instead of using a coarser grid, lower order solution approximation in the finite element space is utilized. Therefore, the same computational grid is used in all approximation level. The *p*-multigrid scheme used in this study is constructed by the following steps [13]:

1. Obtain the solution at the next time level at P_1 approximation order $\widetilde{Q}_{P_1}^{n+1}$,

$$\widetilde{Q}_{P_1}^{n+1}=Q_{P_1}^{n}+\boldsymbol{\mathcal{M}}^{-1}\boldsymbol{\mathcal{R}}(Q_{P_1}^{n});$$

2. Restrict the solution and residual at P_1 approximation level to P_0 approximation level,

$$Q^*_{P_0} = \boldsymbol{I}^{P_0}_{P_1} \widetilde{Q}^{n+1}_{P_1},$$
$$\boldsymbol{\mathcal{R}}_{P_0} = \widetilde{I}^{P_0}_{P_1} \boldsymbol{\mathcal{R}}(\widetilde{Q}^{n+1}_{P_1});$$

3. Obtain the solution at the next time level at P_0 approximation order $\widetilde{Q}^{n+1}_{P_0}$,

$$\widetilde{Q}^{n+1}_{P_0} = Q^*_{P_0} + \boldsymbol{\mathcal{M}}^{-1} \boldsymbol{\mathcal{R}}_{P_0};$$

4. Compute correction term at P_0 approximation order C_{P_0},

$$C_{P_0} = \widetilde{Q}^{n+1}_{P_0} - Q^*_{P_0};$$

5. Update the solution at the next time level at P_1 approximate order $Q^{n+1}_{P_1}$,

$$Q^{n+1}_{P_1} = \widetilde{Q}^{n+1}_{P_1} + \boldsymbol{J}^{P_1}_{P_0} C_{P_0}.$$

Note that $I^{P_0}_{P_1}$, $\widetilde{I}^{P_0}_{P_1}$, and $J^{P_1}_{P_0}$ denote the state restriction operator, the residual restriction operator, and the state prolongation operator, respectively.

2.4 Slope limiter

A very simple slope limiter currently in use for enhancing numerical stability at the shock wave is described [20]. Let us define $p^+ = (1+\epsilon)p_{\max}$ where $p_{\max}$ denotes the maximum of the cell averaged pressure values among the nearby cells sharing the cell interface of current computational cell and ϵ is a small positive constant. Similarly, let us define $p^- = (1-\epsilon)p_{\min}$ where $p_{\min}$ denotes the minimum of the cell averaged pressure values. If the maximum value of the pressure at Gaussian quadrature points becomes larger than p^+, we multiply a common positive constant no greater than unity to those DOFs of dependent variables so that the maximum pressure value at quadrature points agrees with p^+. Similarly, if the minimum value of the pressure at the quadrature points becomes smaller than p^-, we multiply a common positive constant to DOFs so that the minimum pressure value agrees with p^-. We choose ϵ to be 0.01 in this study so that the slope limiter does not work at smooth extrema in the flowfield. We note that the positive constant should not be multiplied to the lowest order DOFs because these DOFs actually represent the cell averaged values of the conservative variables.

The present slope limiter is found sufficient to suppress numerical instability when shock waves appear in the flowfield. With shock waves, a machine-zero convergence is generally difficult to attain. The residual first decreases for several orders of magnitude and then begins to fluctuate. Such example will be shown in the later section. The chosen value of 0.01 for ϵ is rather conservative and it is generally possible to use 0.1-0.3 for both the Euler and the Navier-Stokes calculations in transonic flow regime. For strong expansion region, however, ϵ should be decreased to $\mathcal{O}(0.01)$ and sometimes the same slope limiting procedure is needed to apply to density.

2.5 Parallel algorithm

In this work, we implement the pointwise relaxation implicit method on a parallel computer using the METIS grid partitioning [21] and the Message Passing Interface (MPI) library. In the DG method, information exchange with neighboring cells occurs through the numerical flux function and also the local lifting term for the viscous flux determined at the cell interface. This property is retained in the present implicit method. Therefore, even at the inter-domain boundary, a simple method of assigning DOFs in ghost cells is sufficient for communication between neighboring domains. The procedure for parallelization thus becomes extremely simple.

3 Results and discussions

3.1 Linear advection problem

We first show the computed results for 3D linear advection problem to examine the spatial accuracy and the convergence property of the present scheme for various cell geometries. A unit cube is considered as the computational domain. A sinusoidal wave is coming in from three upwind boundary faces and going out from the rest of the boundary faces. The computational domain is first divided into uniform hexahedra with a constant mesh interval δx. These small hexahedra are further subdivided into either of tetrahedral, prismatic and pyramidal cells. Three different mesh intervals (δx) are considered for examination of spatial accuracy of the present scheme. The number of cells for each computational mesh is summarized in Table 1. In Fig. 2, the error norms for various cell geometries are shown to decrease as the mesh interval is decreased. From the slope of the curve, one can find the actual spatial accuracy of the scheme. For all cell types, the present second order DG scheme well attains the formal accuracy.

Table 1: The number of computational cells for 3D scalar advection problem.

δx	tetrahedron	hexahedron	prism	pyramid
1/10	6,000	1,000	2,000	6,000
1/20	48,000	8,000	16,000	48,000
1/40	432,000	72,000	144,000	432,000

In Fig. 3, the convergence histories of the present implicit scheme are shown with the CFL number of 10^6. One finds that a machine-zero convergence is quickly reached for all cell types.

3.2 Inviscid compressible flowfield over ONERA-M6 isolated wing

Next, we consider the parallel computation of the inviscid compressible flowfield over ONERA-M6 isolated wing as a typical test problem. The parallel computation is per-

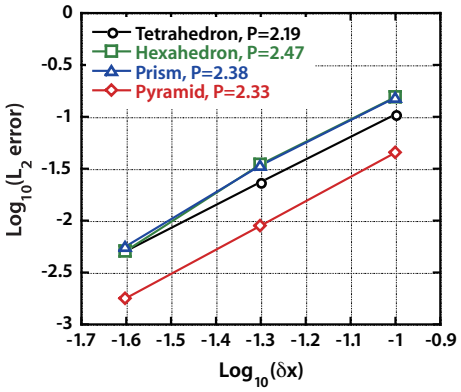


Figure 2: Error norms for various cell geometries are plotted in terms of mesh interval. The slope of the curve gives the spatial accuracy of the present scheme.

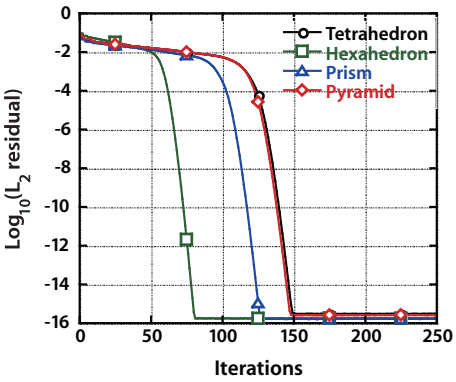


Figure 3: Convergence histories of 3D scalar advection problem in terms of iteration for the case with $\delta x = 1/20$ and $CFL = 10^6$.

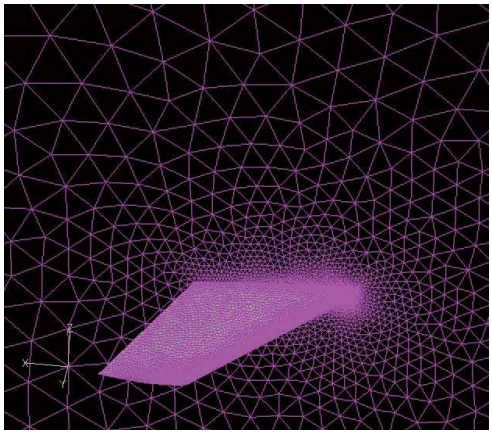


Figure 4: Unstructured mesh for ONERA-M6 isolated wing with 393,979 tetrahedral cells.

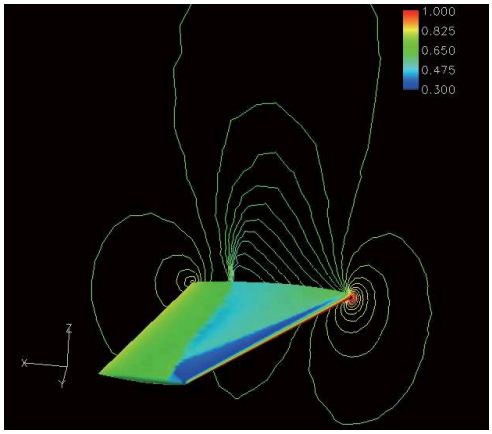


Figure 5: Pressure contours both on the wing upper surface and on the root plane.

formed using up to 128 processors of the SGI Altix 3700Bx2 at the Institute of Fluid Science, Tohoku University. Fig. 4 shows the unstructured mesh for ONERA-M6 isolated wing. It has 393,979 tetrahedral cells. In the calculation, the angle of attack is 3.06 deg, and the freestream Mach number is 0.84. The CFL number is set to be 10^6 and the local time stepping is employed. The computed pressure contours for the converged solution are shown in Fig. 5. One can find that a typical lambda shaped shock pattern appears on the upper surface. The pressure coefficient profiles at 20% and 90% semi-span locations are plotted in Fig. 6 together with the corresponding experimental data [22]. The computed results agree fairly well with the experimental data with small difference caused by viscous effect. Note that numerical wiggles do not appear at the shock wave indicating that the present slope limiter is functioning properly.

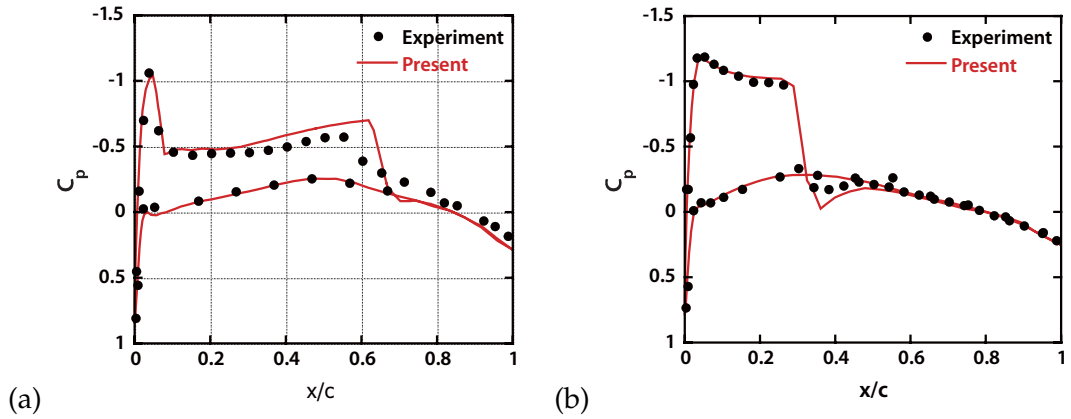


Figure 6: Pressure coefficient (Cp) profiles; (a) at 20% spanwise location, and (b) at 90% spanwise location. Symbols represent the Cp profiles from the experimental data and solid lines represent the computed Cp profiles.

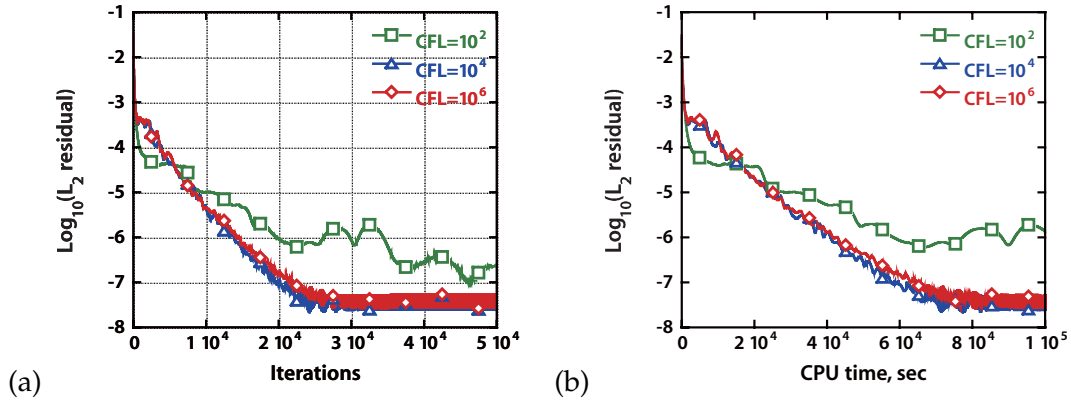


Figure 7: Convergence histories with different CFL numbers for flowfield over ONERA-M6 isolated wing in terms of; (a) number of iterations, and (b) CPU time.

The convergence histories for this problem with different CFL numbers are shown in Fig. 7 in terms of number of iterations and CPU time. The residuals estimated by L_2 norm of the density variation decrease six orders in magnitude and then begin to fluctuate probably due to the use of the present simple slope limiter. As indicated in Fig. 7, the convergence history for CFL number of 10^4 is almost identical with that for 10^6, suggesting that relaxation to steady solution occurs in the same way when the CFL number exceeds a certain value.

The parallel performance is examined in terms of speedup ratio and parallel efficiency. The speedup ratio S_n and the parallel efficiency E_n of the parallel computation using n processors are defined, respectively, as

$$S_n = \frac{T_1}{T_n}, \tag{3.1}$$

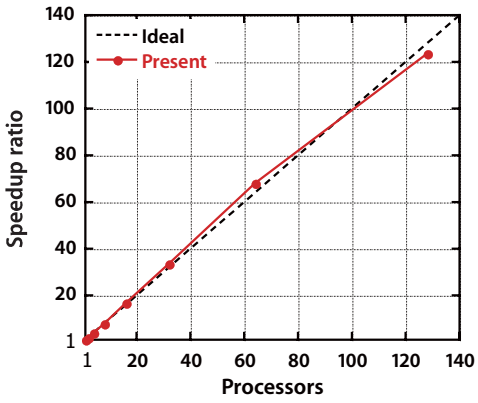


Figure 8: Speedup ratio. Dashed line represents the ideal speedup ratio, and solid line with symbols represents the obtained speedup ratio.

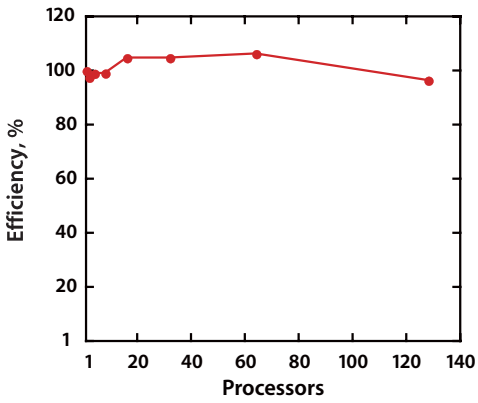
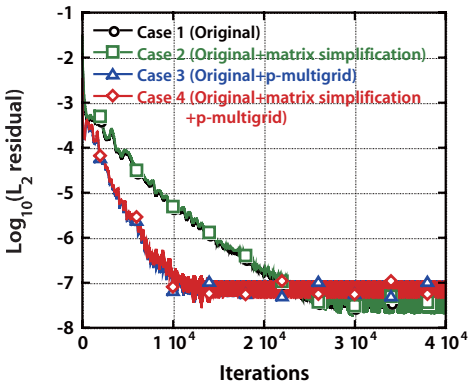
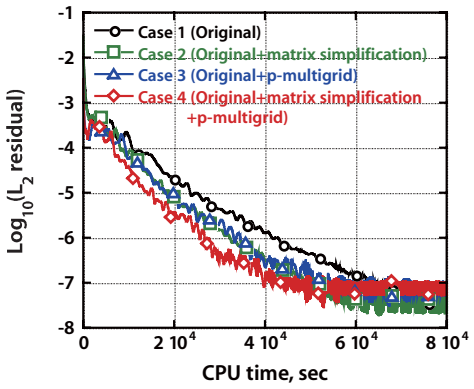


Figure 9: Obtained parallel efficiency.



(a)



(b)

Figure 10: Convergence histories for the flowfield over ONERA-M6 isolated wing in terms of; (a) number of iterations, and (b) CPU time.

and

$$E_n = \frac{T_1}{n \times T_n} \times 100[\%], \tag{3.2}$$

where T_1 denotes the wall-clock time of serial execution and T_n the wall-clock time of parallel execution using n processors. The obtained speedup ratio and the parallel efficiency are plotted in Figs. 8 and 9, respectively. A super linear speedup is observed for 64 processors. Even up to 128 processors, one can find that the speedup ratio is almost scalable and the sustained parallel efficiency is fairly high. These evidences indicate that the present pointwise relaxation implicit DG method can readily exploit the full performance of parallel computers.

We then examine the improved computational efficiency brought by the matrix simplification given by Eq. (2.35) and the p-multigrid scheme. For this comparison, all the computations are carried out using 8 processors. Table 2 summarizes the test cases in

Table 2: Time integration schemes.

Case	Numerical methods
1	Pointwise relaxation implicit scheme
2	Pointwise relaxation implicit scheme + matrix simplification
3	Pointwise relaxation implicit scheme + p-multigrid
4	Pointwise relaxation implicit scheme + matrix simplification + p-multigrid

which the baseline implicit scheme is combined with the matrix simplification and the p-mutigrid scheme. The convergence histories of these cases in terms of number of iterations and also of CPU time are shown in Figs. 10(a) and 10(b), respectively. One can say that the matrix simplification does not influence the convergence characteristics. The number of iterations to reach convergence is halved when the p-multigrid scheme is used. The CPU time to reach convergence is almost the same for case 2 and case 3, and is about $2/3$ of that for case 1. By combining both the matrix simplification and the p-multigrid scheme, the overall CPU time to reach convergence is $1/2$ of that for case 1. Therefore, one can say that a combination of the matrix simplification and the p-multigrid is quite effective in reducing computational costs. We note that the parallel efficiency is found to be unchanged for all cases listed in Table 2.

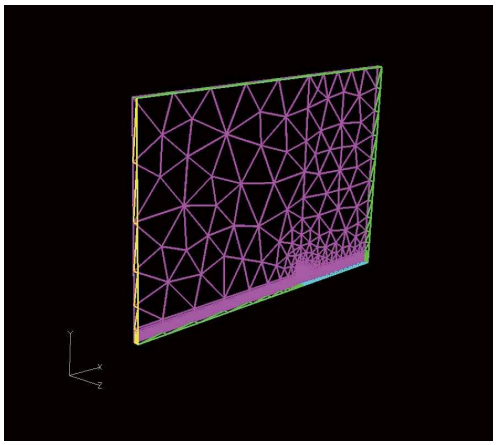


Figure 11: Hybrid mesh for the laminar boundary layer over a flat plate.

3.3 Laminar boundary layer flowfield over flat plate

In this subsection, we will show the computed results for the laminar boundary layer flow over a flat plate. Fig. 11 shows the hybrid mesh for the flat plate. It has 2,183 tetrahedral cells and 3,600 prismatic cells. The freestream Mach number is chosen to be 0.5 and the Reynolds number is 10^7. The CFL number for the implicit integration is 10^5

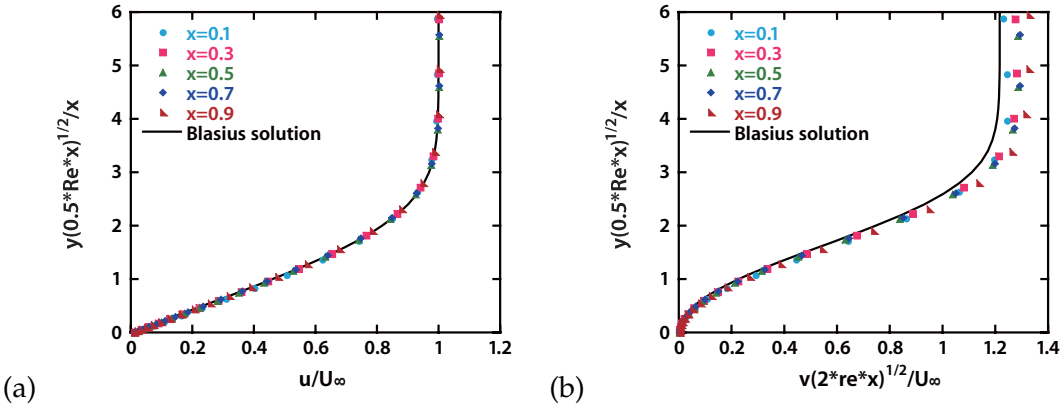


Figure 12: Velocity profiles of; (a) x direction, and (b) y direction.

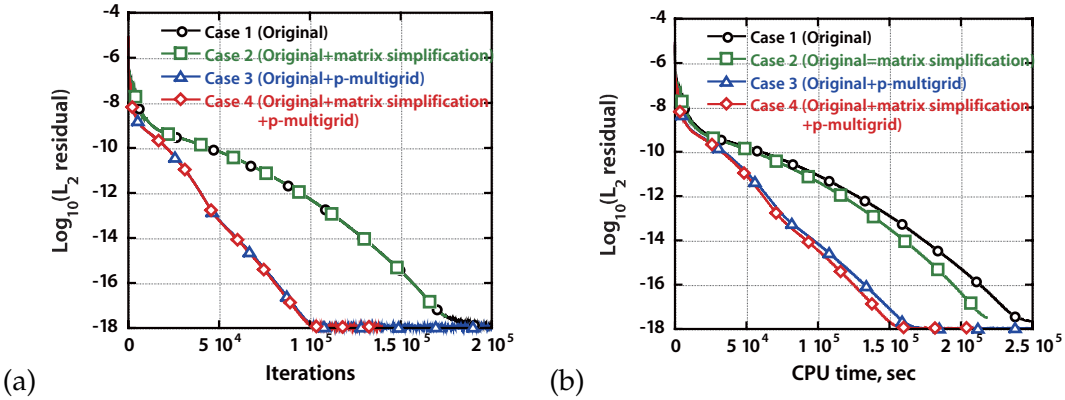


Figure 13: Convergence histories for the laminar boundary layer flowfield over a flat plate in terms of; (a) number of iterations, and (b) CPU time.

and the local time stepping is employed. All the computations in this subsection and in the next subsection are carried out using Xeon dual core 3.2GHz×2 (4 cores).

The computed velocity profiles are plotted with the Blasius solutions in Fig. 12. As can be seen, a good agreement with the Blasius solution is obtained for x component. As to y component, a reasonable agreement is obtained, though some differences with the Blasius solution appear particularly at the boundary layer edge. The convergence histories for this problem are shown in Fig. 13. A machine zero convergence is obtained for all the cases. The matrix simplification achieves 10% reduction of the total CPU time of the original implicit method to reach convergence, while the convergence history is virtually the same as that of the original method. The matrix simplification is less effective for this case because the total amount of arithmetic operations is sharply increased due to evaluation of viscous terms and lifting terms. When the p-multigrid scheme is combined with the original method, the number of iterations needed for convergence becomes half

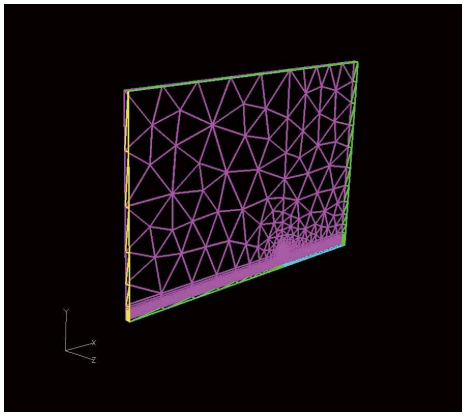


Figure 14: Hybrid mesh for the turbulent boundary layer over a flat plate.

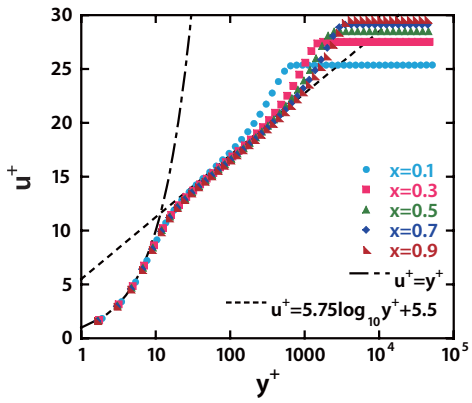


Figure 15: u^+ distribution in the turbulent boundary layer.

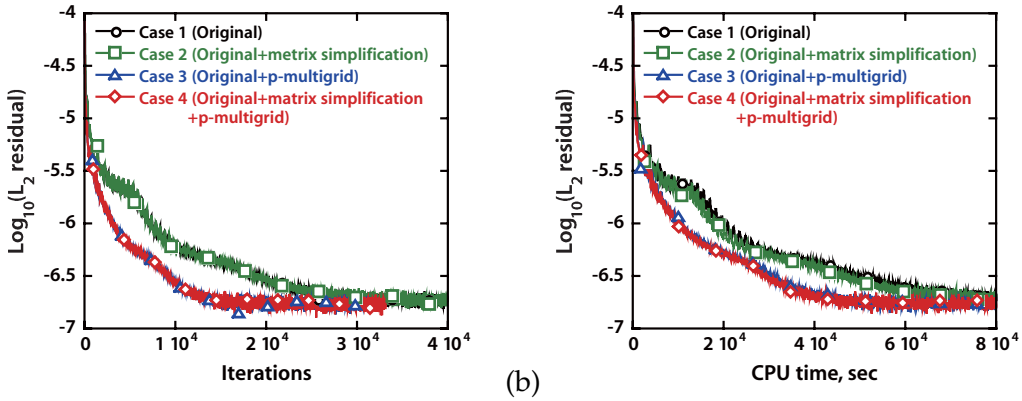


Figure 16: Convergence histories for the turbulent boundary layer flowfield over a flat plate in terms of; (a) number of iterations, and (b) CPU time.

of that for the original method and the CPU time becomes 3/5. The combination of the matrix simplification and the p-multigrid is shown to reduce 5% of the total CPU time of that given by the p-multigrid alone.

3.4 Turbulent boundary layer flowfield over flat plate

We now examine the turbulent boundary layer flowfield over a flat plate. The hybrid mesh for the flat plate, which consists of 3,353 tetrahedral cells and 8,000 prismatic cells, is shown in Fig. 14. The freestream Mach number is 0.5, and the Reynolds number is 10^7. The CFL number is 10^5 and the local time stepping is employed.

The computed velocity profiles are plotted at several streamwise locations in Fig. 15. A good agreement with the log law is obtained. Fig. 16 shows the convergence histories of the present methods. A machine zero convergence is not achieved for this problem

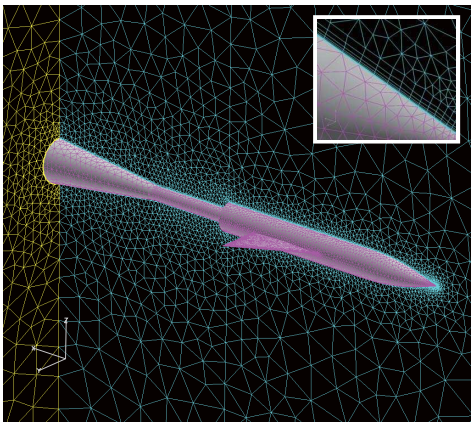


Figure 17: Unstructured hybrid mesh for AGARD-B model with 198,131 tetrahedral cells and 158,070 prismatic cells.

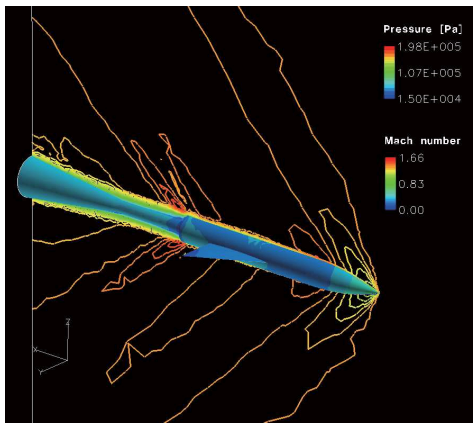


Figure 18: Pressure contours on the surface and Mach number contours on the root plane.

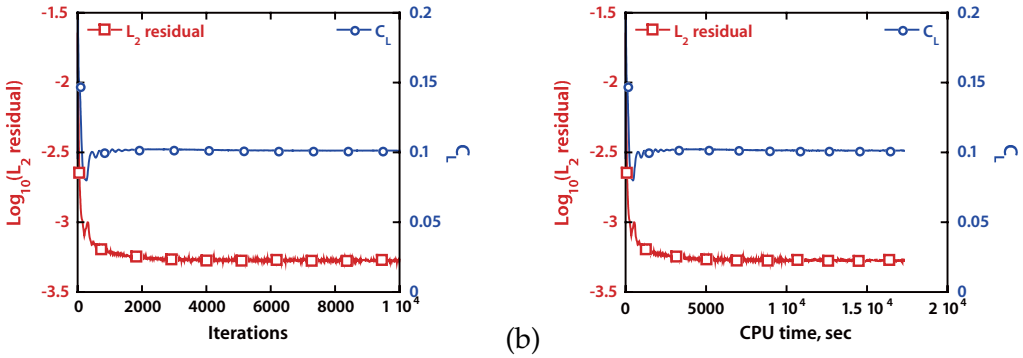


Figure 19: Convergence history for the flowfield over AGARD-B model in terms of; (a) number of iterations, and (b) CPU time.

due to fluctuation of computed eddy viscosity in the boundary layer. As in the laminar case, a 10% reduction of the CPU time is attained with the matrix simplification, while the convergence history is the same as that for the original method. Again, by combining p-multigrid scheme with the original method, the number of iterations needed for convergence becomes half of that for the original method, and the CPU time becomes 3/5. The combination of the matrix simplification and the p-multigrid is found to be less effective for this problem.

3.5 RANS simulation over AGARD-B wind tunnel calibration model

We finally show the computed results for the viscous compressible flowfield over AGARD-B wind tunnel calibration model. This computation is also performed using 128 processors on the SGI Altix where the original implicit method is used as a baseline computation. Fig. 17 shows the unstructured hybrid mesh for AGARD-B model. It has

198,131 tetrahedral cells, and 158,070 prismatic cells. The angle of attack is 2.2 deg. The freestream Mach number is 1.4, and the Reynolds number is 2.8×10^7/m. The CFL number is set to be 10^5 and the local time stepping is again used in the calculation.

The computed pressure contours on the model surface together with the Mach number contours plotted on the root plane are shown in Fig. 18. Although the shock wave is smeared out due to use of coarser mesh, the characteristic flow patterns are well obtained. The obtained convergence histories are shown in Fig. 19. Both L_2 residual and the lift coefficient are converged within 3,000 iterations and 1.5×10^5 sec, respectively. However, the L_2 residual only decreases for two orders of magnitude for this problem. This is caused by the fluctuations of the computed flowfield at the boat tail of the fuselage where the boundary layer is separated from the edge. The computed lift coefficient becomes 0.1013, while that of the experimental data is 0.1000. Although a reasonable agreement is obtained for the lift coefficient with the experimental data, the computed drag coefficient becomes almost 30% higher than the experimental data due to coarser mesh not only at the wing leading edge but also at the forebody region. A more detailed computation using a much finer mesh is certainly needed for obtaining a quantitative agreement of drag coefficient.

4 Concluding remarks

The pointwise relaxation implicit DG method is developed. By applying several convergence acceleration methods, the number of iterations and the total CPU time of the DG method needed for convergence are reduced both for the inviscid compressible flowfield and for the viscous compressible flowfield. It is also shown that the present implicit scheme can readily achieve scalable speedup ratio and fairly good parallel efficiency, while maintaining numerical stability and favorable convergence property. It is therefore shown in the present study that the pointwise relaxation implicit algorithm can enhance the applicability of the DG method for various practical problems, for which conventional DG methods cannot be employed due to extremely high computational cost.

Acknowledgments

The parallel computations were carried out using SGI Altix 3700Bx2 at the Institute of Fluid Science, Tohoku University.

References

[1] T. J. Barth, Aspects of unstructured grids and finite-volume solvers for the Euler and Navier-Stokes equations, AGARD-R-787, (1992), 6-1-6-61.
[2] X-D. Liu, S. Osher, and T. Chan, Weighted essentially non-oscillatory schemes, J. Comput. Phys., 115 (1994), 200-212.

[3] B. Cockburn, and C-W. Shu, TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws II: General framework, Math. Comp., 52 (1989), 411-435.
[4] B. Cockburn, and C-W. Shu, The Runge-Kutta discontinuous Galerkin method for conservation laws V: Multidimensional systems, J. Comput. Phys., 141 (1998), 199-224.
[5] F. Bassi, and S. Rebay, A High Order Discontinuous Galerkin Method for Compressible Turbulent Flow. in B. Cockburn, G. E. Karniadakis, and C.-W. Shu, editors, Discontinuous Galerkin Method: Theory, Computations and Applications, Lecture Notes in Computational Science and Engineering 11, 113-123, Springer-Verlag, 2000.
[6] P. Rasetarinera, and M. Y. Hussaini, An efficient implicit discontinuous spectral Galerkin method, J. Comput. Phys., 172, (2001), 718-738.
[7] R. Hartmann, Adaptive discontinuous Galerkin methods with shock-capturing for the compressible Navier-Stokes equations, Int. J. Numer. Meth. Fluids, 51, (2006), 1131-1156.
[8] R. Hartmann, and P. Houston, Symmetric interior penalty DG methods for the compressible Navier-Stokes equations I: Method formulation, Int. J. Numer. Anal. Model., 3, (2006), 1-20.
[9] R. Hartmann, and P. Houston, Symmetric interior penalty DG methods for the compressible Navier-Stokes equations II: Goal-oriented a posteriori error estimation, Int. J. Numer. Anal. Model., 3, (2006), 141-162.
[10] V. Dolejší, and M. Feistauer, A semi-implicit discontinuous Galerkin finite element method for the numerical solution of inviscid compressible flow, J. Comput. Phys., 198, (2004), 727-746.
[11] V. Dolejší, Semi-implicit interior penalty discontinuous Galerkin methods for viscous compressible flows, Commun. Comput. Phys., 4, (2008), 231-274.
[12] C. R. Nastase, and D. J. Mavriplis, High-order discontinuous Galerkin methods using an hp-multigrid approach, J. Comput. Phys., 213, (2006), 330-357.
[13] H. Luo, J. D. Baum, and R. Löhner, Fast p-multigrid discontinuous Galerkin method for compressible flow at all speeds, AIAA J., 46 (2008), 635-652.
[14] S. J. Sherwin, and G. E. Karniadakis, A new triangular and tetrahedral basis for high-order (hp) finite element methods, Int. J. Num. Meth. Eng., 38 (1995), 3775-3802.
[15] M. Dubiner, Spectral methods on triangles and other domains, J. Sci. Comp., 6 (1991), 345-390.
[16] Y. Wada, and M-S. Liou, A flux splitting scheme with high-resolution and robustness for discontinuities, AIAA Paper 94-0083, (1994).
[17] F. Bassi, and S. Rebay, Numerical evaluation of two discontinuous Galerkin methods for the compressible Navier-Stokes equations, Int. J. Numer. Meth. Fluids, 40, (2002), 197-207.
[18] P. R. Spalart, and S. R. Allmaras, A one-equation turbulence model for aerodynamic flows, AIAA Paper 92-0439, (1992).
[19] A. Jameson, and S. Yoon, Lower-upper implicit schemes with multiple grids for the Euler equations, AIAA J., 25, (1987), 929-935.
[20] T. Haga, and K. Sawada, An improved slope limiter for high-order spectral volume methods solving the 3D compressible Euler equations, in preparation, (2009).
[21] G. Karypis, and V. Kumar, A fast and high quality multilevel scheme for partitioning irregular graphs, SIAM J. Sci. Comp., 20, (1998), 359-392.
[22] V. Shumitt, and F. Charpin, Pressure distributions on the ONERA-M6-wing at transonic Mach numbers, AGARD AR-138-B1 (1979).