Springer

Acta Mechanica Sinica

CrossMark click for updates

• RESEARCH PAPER •

Acta Mech. Sin., Vol. 40, 323429 (2024)
https://doi.org/10.1007/s10409-024-23429-x

A robust implicit high-order discontinuous Galerkin method for solving compressible Navier-Stokes equations on arbitrary grids

Jia Yan[1], Xiaoquan Yang[1*], and Peifen Weng[1,2]

[1] *Shanghai Institute of Applied Mathematics and Mechanics, Shanghai Key Laboratory of Mechanics in Energy Engineering, School of Mechanics and Engineering Science, Shanghai University, Shanghai 200072, China;*
[2] *College of Energy and Mechanical Engineering, Shanghai University of Electric Power, Shanghai 200090, China*

Received November 19, 2023; accepted December 15, 2023; published online June 11, 2024

The primary impediments impeding the implementation of high-order methods in simulating viscous flow over complex configurations are robustness and convergence. These challenges impose significant constraints on computational efficiency, particularly in the domain of engineering applications. To address these concerns, this paper proposes a robust implicit high-order discontinuous Galerkin (DG) method for solving compressible Navier-Stokes (NS) equations on arbitrary grids. The method achieves a favorable equilibrium between computational stability and efficiency. To solve the linear system, an exact Jacobian matrix solving strategy is employed for preconditioning and matrix-vector generation in the generalized minimal residual (GMRES) method. This approach mitigates numerical errors in Jacobian solution during implicit calculations and facilitates the implementation of an adaptive Courant-Friedrichs-Lewy (CFL) number increasing strategy, with the aim of improving convergence and robustness. To further enhance the applicability of the proposed method for intricate grid distortions, all simulations are performed in the reference domain. This practice significantly improves the reversibility of the mass matrix in implicit calculations. A comprehensive analysis of various parameters influencing computational stability and efficiency is conducted, including CFL number, Krylov subspace size, and GMRES convergence criteria. The computed results from a series of numerical test cases demonstrate the promising results achieved by combining the DG method, GMRES solver, exact Jacobian matrix, adaptive CFL number, and reference domain calculations in terms of robustness, convergence, and accuracy. These analysis results can serve as a reference for implicit computation in high-order calculations.

Discontinuous Galerkin method, Exact Jacobian matrix, GMRES solver, Adaptive CFL number, Reference domain, High-order

Citation: J. Yan, X. Yang, and P. Weng, A robust implicit high-order discontinuous Galerkin method for solving compressible Navier-Stokes equations on arbitrary grids, Acta Mech. Sin. 40, 323429 (2024), https://doi.org/10.1007/s10409-024-23429-x

1. Introduction

Discontinuous Galerkin (DG) methods have gained widespread utilization in computational fluid dynamics (CFD), computational acoustics, and computational magnetohydrodynamics [1-5]. They offer appealing characteristics such as robustness and accuracy in high-order calculations, surpassing finite element (FE) and finite volume (FV) methods.

In CFD, DG methods have proven successful and effective in handling compressible Euler equations at high-order accuracy [6-8]. However, when extending these methods to calculations involving diffusion problems, such as solving compressible Navier-Stokes (NS) equations and Reynolds-averaged Navier-Stokes (RANS) equations, the robustness and efficiency of DG methods still pose a challenge. Nonetheless, several different ways of handling the diffusion terms in DG methods have been proposed, including the symmetric interior penalty (SIP) [9], local discontinuous Galerkin (LDG) [10], compact discontinuous Galerkin

*Corresponding author. E-mail address: quanshui@shu.edu.cn (Xiaoquan Yang)
Executive Editor: Zhenhua Xia

(CDG) [11], recovery-based discontinuous Galerkin (RDG) [12], reconstructed discontinuous Galerkin (rDG) [13], and the second Bassi-Rebay (BR2) [14] schemes. Besides, a direct discontinuous Galerkin (DDG) method introduced by Liu and Yan [15,16] shows its promising as an alternative choice in handling viscous terms. The primary characteristic of the DDG method lies in the incorporation of a numerical flux, which effectively approximates the gradient of the solution across the edges of the elements. Later, Cheng et al. [17,18] extended the DDG method for solving the NS equations and RANS equations on arbitrary grids, with the consideration of spatial discretization up to order $p = 2$. Our goal here is to present NS and RANS solutions using DDG method with higher-order polynomial approximations in the range p = 3-5.

To attain our objective, it is imperative to mitigate the impact of the grid on the calculations. It is widely acknowledged that turbulent simulation boundary layer grids exhibit a substantial aspect ratio, and unstructured grids are susceptible to large distortion. This characteristic is manifested in the DG method, resulting in increased rigidity of the mass matrix, particularly in higher-order DG methods. As the order increases, the size of the mass matrix expands, magnifying the impact of the grid on the reversibility of the mass matrix. The poor grid quality will directly lead to the irreversibility of the mass matrix, consequently impacting the stability of the calculation. Two approaches can be employed to address this predicament. The first approach involves increasing the number of grids, ensuring grid quality but incurring higher computational costs. The second approach involves transforming the calculation domain into the reference domain [19,20], thereby reducing the grid quality dependence. This technique holds great potential for solving RANS equations utilizing high-order method.

The robustness and stability of implicit schemes also play a very important role in high-order calculations. The generalized minimal residual (GMRES) method with preconditioner [21-24] emerges as an efficient implicit scheme for solving large sparse linear systems [25]. It exhibits superior stability and convergence, rendering it the most commonly employed method in DG solution [26-28]. However, most existing studies merely employ approximate Jacobian matrices for implicit schemes [29-31], derived purely from inviscid flux or inviscid flux combined with the viscous spectral radius. This approach fails to account for the exact behavior of laminar and turbulent flows, introducing errors that lead to significant declines in computational efficiency and stability during high-order calculations. Fortunately, the Jacobian matrix of DG discretization can be derived exactly using the chain rule. The adoption of this exact Jacobian matrix solving strategy [32,33] has demonstrated excellent performance in practical applications. Moreover, the robustness and efficiency of the GMRES method for high-order solutions are heavily influenced by various parameters [27,34-36], such as the value of the Courant-Friedrichs-Lewy (CFL) number, preconditioner selection, accuracy of preconditioning and matrix-vector generation, Krylov subspace settings, and convergence criteria of GMRES. Properly combining these factors further enhances the computational efficiency and stability of the GMRES method. While little work has been done to explore the effects of these parameters on higher-order DG calculations.

Considering the aforementioned discussions, this paper presents a robust implicit high-order DG method designed to solve compressible NS and RANS equations on arbitrary grids. The DDG scheme is employed to discretize the viscous flux, while the Roe scheme is utilized for discretizing the inviscid flux. Turbulent flows are solved using the negative SA turbulence model coupled with RANS equations. The current implicit method incorporates three key strategies to enhance the robustness and convergence of calculations. Firstly, solutions are transformed into a reference domain, reducing dependency on computational grids. Secondly, an exact Jacobian matrix is employed without any approximation or simplification when solving the linear system of equations using the GMRES method, utilizing the lower-upper symmetric Gauss-Seidel (LU-SGS) preconditioner. Finally, an adaptive CFL number increasing strategy is implemented to achieve a better balance between stability and efficiency as this strategy is able to dynamically adjust the CFL number in response to changes in the flow field. These strategies facilitate the extension of the method for higher-order calculations. Numerous benchmark test cases, including both laminar and turbulent flows, are selected to evaluate the performance of this method. In addition, a comprehensive analysis of various parameters that affect computational stability and efficiency is conducted. The computed results serve as a reference for implicit computation in high-order DG calculations and highlight the promising features and potential of the proposed method, particularly regarding its robustness and efficiency.

2. Governing equations

The conservative formulation of the compressible RANS equations, coupled with the one-equation negative SA turbulence model, can be expressed as follows:

$$\frac{\partial \mathbf{U}}{\partial t}+\frac{\partial \mathbf{F}_i}{\partial x_i}-\frac{\partial \mathbf{G}_i}{\partial x_i}=\mathbf{S}. \tag{1}$$

In Eq. (1), the conservative variables $\mathbf{U}$, convective flux $\mathbf{F}_i$, diffusive flux $\mathbf{G}_i$, and source term $\mathbf{S}$ are defined by

$$\mathbf{U}=\begin{bmatrix}\rho\\ \rho u_1\\ \rho u_2\\ \rho e\\ \rho\tilde{\upsilon}\end{bmatrix},\qquad \mathbf{F}_i=\begin{bmatrix}\rho u_i\\ \rho u_i u_1+p\delta_{i1}\\ \rho u_i u_2+p\delta_{i2}\\ \rho h u_i\\ \rho u_i\tilde{\upsilon}\end{bmatrix},$$

$$\mathbf{G}_i=\begin{bmatrix}0\\ \sigma_{i1}\\ \sigma_{i2}\\ u_j\sigma_{ij}+k\dfrac{\partial T}{\partial x_i}\\ \dfrac{(\mu_1+\rho\tilde{\upsilon}f_n)}{\sigma}\dfrac{\partial\tilde{\upsilon}}{\partial x_i}\end{bmatrix},\quad \mathbf{S}=\begin{bmatrix}0\\ 0\\ 0\\ 0\\ -\rho(P-D)-\dfrac{1}{\sigma}\nabla\tilde{\upsilon}\cdot\nabla\tilde{\upsilon}\end{bmatrix}, \tag{2}$$

where u_1, u_2, x_1, x_2 are noted as u, v, x, y. The pressure p is determined by the equation of state of ideal gas law,

$$p=(\gamma-1)\left[\rho e-\frac{\rho}{2}(u^2+v^2)\right],\quad \rho h=\rho e+p, \tag{3}$$

with γ equals to 1.4, δ_{ij} represents the Kronecker delta and k is the thermal conductivity of the fluid. The components of the stress tensor are given by

$$\sigma_{ij}=(\mu_1+\mu_t)\left(\frac{\partial u_i}{\partial x_j}+\frac{\partial u_j}{\partial x_i}-\frac{2}{3}\delta_{ij}\frac{\partial u_k}{\partial x_k}\right). \tag{4}$$

The turbulence production and destruction terms, P and D are

$$P=\begin{cases}c_{b1}(1-f_{t2})\tilde{S}\tilde{\upsilon}, & \tilde{\upsilon}\geqslant 0,\\ c_{b1}(1-c_{t3})S\tilde{\upsilon}, & \tilde{\upsilon}<0,\end{cases}$$

$$D=\begin{cases}\left(c_{w1}f_w-\dfrac{c_{b1}f_{t2}}{k^2}\right)\left(\dfrac{\tilde{\upsilon}}{d}\right)^2, & \tilde{\upsilon}\geqslant 0,\\ -c_{w1}\left(\dfrac{\tilde{\upsilon}}{d}\right)^2, & \tilde{\upsilon}<0,\end{cases} \tag{5}$$

where μ_t denotes the turbulence eddy viscosity

$$\mu_t=\begin{cases}\rho\tilde{\upsilon}f_{\upsilon 1}, & \tilde{\upsilon}\geqslant 0,\\ 0, & \tilde{\upsilon}<0.\end{cases} \tag{6}$$

The modified vortices magnitude, $\tilde{S}$ is given by

$$\tilde{S}=\begin{cases}S+\bar{S}, & \bar{S}>c_{v2}S,\\ S+\dfrac{S\left(c_{v2}^2+c_{v3}\bar{S}\right)}{(c_{v3}-2c_{v2})S-\bar{S}}, & \bar{S}<c_{v2}S.\end{cases} \tag{7}$$

Other parameters are given as follows:

$$\begin{aligned}&S=\sqrt{\omega_{ij}\omega_{ij}}, && \bar{S}=\frac{\mu_1\chi}{\rho k^2 d^2}f_{v2},\\ &f_{\upsilon 1}=\frac{\chi^3}{\chi^3+c_{v1}^3}, && f_{\upsilon 2}=1-\frac{\chi}{1+\chi f_{\upsilon 1}},\\ &\chi=\frac{\rho\tilde{\upsilon}}{\mu_1}, && r=\min\left[\frac{\mu_1\chi}{\rho\tilde{S}\kappa^2 d^2},10\right],\\ &g=r+c_{w2}(r^6-r), && f_w=g\left(\frac{1+c_{w3}^6}{g^6+c_{w3}^6}\right)^{1/6},\end{aligned} \tag{8}$$

here, $\omega_{ij}=0.5\left(\partial u_i/\partial x_j-\partial u_i/\partial x_j\right)$ pertains to the rotation tensor and d denotes the distance to the closest wall. The specific values for the constants can be found in Ref. [37]. The no-slip boundary conditions are imposed on the surfaces in contact with the wall. For all test cases, the free-stream turbulent working variable is uniformly set as $\rho\tilde{\upsilon}=4\rho\tilde{\upsilon}_\infty$.

3. DG formulation on arbitrary grid

3.1 DG formulation for RANS-SA equations

To discretize the governing equation stated in Eq. (1), a DG formulation is employed. Firstly, we introduce the subsequent weak form by multiplying the conservation law above with a test function W [1], followed by integration by parts over the entire domain.

$$\begin{aligned}&\frac{\mathrm{d}}{\mathrm{d}t}\int_\Omega \mathbf{U}_h W_h\mathrm{d}\Omega+\int_\Gamma \mathbf{F}_k\mathbf{n}_k W_h\mathrm{d}\Gamma-\int_\Omega \mathbf{F}_k\frac{\partial W_h}{\partial x_k}\mathrm{d}\Omega\\ &=\int_\Gamma \mathbf{G}_k\mathbf{n}_k W_h\mathrm{d}\Gamma-\int_\Omega \mathbf{G}_k\frac{\partial W_h}{\partial x_k}\mathrm{d}\Omega+\int_\Omega \mathbf{S}W_h\mathrm{d}\Omega,\end{aligned} \tag{9}$$

where Γ denotes the boundary of the cell Ω and $\mathbf{n}_k$ denotes the unit outward normal vector to the boundary, We assume that the domain Ω is subdivided into a collection of non-overlapping elements Ω_e. We introduce the following broken Sobolev space $\mathbf{V}_h^p$,

$$\mathbf{V}_h^p=\left\{\mathbf{v}_h\in\left[L_2(\Omega)\right]^m:\mathbf{v}_h\mid\Omega_e\in\left[\mathbf{V}_p^m\right]\forall\,\Omega_e\in\Omega\right\}, \tag{10}$$

which consists of discontinuous vector-values polynomial functions of degree p, where m is the dimension of the unknown vector and

$$\mathbf{V}_p^m=\mathrm{span}\left\{\prod x_i^{\alpha_i}:0\leq\alpha\leq p,\ 0\leq i\leq d\right\}, \tag{11}$$

where α denotes a multi-index and d is the dimension of space. Then, it can obtain the following semi-discrete form by applying weak formulation on each element Ω_e, find $\mathbf{U}_h\in\mathbf{V}_h^p$, such as

$$\begin{aligned}&\frac{\mathrm{d}}{\mathrm{d}t}\int_{\Omega_e}\mathbf{U}_h W_h\mathrm{d}\Omega+\int_{\Gamma_e}\mathbf{F}_k(\mathbf{U}_h)\mathbf{n}_k W_h\mathrm{d}\Gamma-\int_{\Omega_e}\mathbf{F}_k(\mathbf{U}_h)\frac{\partial W_h}{\partial x_k}\mathrm{d}\Omega\\ &=\int_{\Gamma_e}\mathbf{G}_k\left(\mathbf{U}_h,\nabla\mathbf{U}_h\right)\mathbf{n}_k W_h\mathrm{d}\Gamma-\int_{\Omega_e}\mathbf{G}_k\left(\mathbf{U}_h,\nabla\mathbf{U}_h\right)\frac{\partial W_h}{\partial x_k}\mathrm{d}\Omega\\ &\quad+\int_{\Omega_e}\mathbf{S}\left(\mathbf{U}_h,\nabla\mathbf{U}_h\right)W_h\mathrm{d}\Omega,\end{aligned} \tag{12}$$

where $\mathbf{U}_h$, W_h represent the FE approximations to the analytical solution $\mathbf{U}$ and the test function W respectively. They are approximated by a piecewise polynomial function of degree p and are discontinuous at the cell interfaces.

3.2 DG formulation in global domain for RANS-SA system

In global domain, assume that B is the basis of polynomial function of degree p, this is then equivalent to the following system of N equations

$$\begin{aligned}&\frac{\mathrm{d}}{\mathrm{d}t}\int_{\Omega_e}\mathbf{U}_h B_i \mathrm{d}\Omega+\int_{\Gamma_e}\mathbf{F}_k\left(\mathbf{U}_h^{\mathrm{L}},\mathbf{U}_h^{\mathrm{R}}\right)\mathbf{n}_k B_i \mathrm{d}\Gamma-\int_{\Omega_e}\mathbf{F}_k(\mathbf{U}_h)\frac{\partial B_i}{\partial x_k}\mathrm{d}\Omega\\&=\int_{\Gamma_e}\mathbf{G}_k\left(\mathbf{U}_h^{\mathrm{L}},\nabla\mathbf{U}_h^{\mathrm{L}},\mathbf{U}_h^{\mathrm{R}},\nabla\mathbf{U}_h^{\mathrm{R}}\right)\mathbf{n}_k B_i \mathrm{d}\Gamma-\int_{\Omega_e}\mathbf{G}_k(\mathbf{U}_h)\frac{\partial B_i}{\partial x_k}\mathrm{d}\Omega\\&\quad+\int_{\Omega_e}\mathrm{S}\left(\mathbf{U}_h,\nabla\mathbf{U}_h\right)B_i \mathrm{d}\Omega,\end{aligned} \tag{13}$$

where N represents the dimension of the polynomial space, while the conservative state vectors at the left and right interfaces of the element are denoted by $\mathbf{U}_h^{\mathrm{L}}$ and $\mathbf{U}_h^{\mathrm{R}}$, respectively. The DG method of degree p, also termed as DG (P) method, closely resembles FV schemes in its utilization of numerical fluxes. It should be emphasized that the classical first-order cell-centered FV scheme is an exact correspondence to the DG(P0) method, which utilizes piecewise constant polynomials. Consequently, DG(Pk) methods with $k > 0$ can be considered as a natural extension of FV methods to higher-order approaches. By increasing the degree p of the polynomials, the corresponding DG methods of higher order can be obtained.

In the global domain DG methods, the numerical polynomial solutions $\mathbf{U}_h$ within each element are represented using either a standard Lagrange basis or a hierarchical node-based basis, as depicted below

$$\mathbf{U}_h=\sum_{i=1}^{N}\mathbf{U}_i(t)B_i(x), \tag{14}$$

where $B_i(x)$ are the basis functions. Generally, the numerical polynomial solutions are represented using a Taylor series expansion at the centroid of the cell, which is introduced by Luo et al. [1]. As an example, $\mathbf{U}_h$ of P2 is given below

$$\begin{aligned}\mathbf{U}_h=&\overline{\mathbf{U}}B_1+\left.\frac{\partial\mathbf{U}}{\partial x}\right|_c\Delta x B_2+\left.\frac{\partial\mathbf{U}}{\partial y}\right|_c\Delta y B_3+\left.\frac{\partial^2\mathbf{U}}{\partial x^2}\right|_c\Delta x^2 B_4\\&+\left.\frac{\partial^2\mathbf{U}}{\partial y^2}\right|_c\Delta y^2 B_5+\left.\frac{\partial^2\mathbf{U}}{\partial x\partial y}\right|_c\Delta x\Delta y B_6,\end{aligned} \tag{15}$$

with

$$\begin{aligned}&B_1=1,\ B_2=\frac{(x-x_c)}{\Delta x},\ B_3=\frac{(y-y_c)}{\Delta y},\\&B_4=\frac{(x-x_c)^2}{2\Delta x^2}-\frac{1}{\Omega_e}\int_{\Omega_e}\frac{(x-x_c)^2}{\Delta x^2}\mathrm{d}\Omega,\\&B_5=\frac{(y-y_c)^2}{2\Delta y^2}-\frac{1}{\Omega_e}\int_{\Omega_e}\frac{(y-y_c)^2}{\Delta y^2}\mathrm{d}\Omega,\\&B_6=\frac{(x-x_c)(y-y_c)}{\Delta x\Delta y}-\frac{1}{\Omega_e}\int_{\Omega_e}\frac{(x-x_c)(y-y_c)}{\Delta x\Delta y}\mathrm{d}\Omega,\end{aligned} \tag{16}$$

where $\overline{\mathbf{U}}$ is the cell-averaged values. $\Delta x=0.5(x_{\max}-x_{\min})$, $\Delta y=0.5(y_{\max}-y_{\min})$, and $x_{\min}$, $x_{\max}$, $y_{\min}$, $y_{\max}$ are the minimum and maximum coordinates in the cell Ω_e in x and y directions, respectively.

Considering that the control volume remains constant in time, the remaining term in Eq. (13) can be shifted to the right-hand side of the equation, resulting in the semi-discrete form of Eq. (13).

$$\mathbf{M}\frac{\mathrm{d}\mathbf{U}}{\mathrm{d}t}=\mathbf{R}, \tag{17}$$

where $\mathbf{M}$ denotes the mass matrix, $\mathbf{U}$ is the solution vector, and $\mathbf{R}$ is the residual including inviscid flux, viscous flux and their corresponding domain integral term, also the source term. The specific formulas of $\mathbf{M}$ are written as

$$\mathbf{M}=\int_{\Omega_e}B_i(x)B_j(x)\mathrm{d}\Omega. \tag{18}$$

According to the expression in Eq. (18), the size of the mass matrix increases as higher-order basis functions with more degrees are employed, leading to increased stiffness. This, in turn, reduces reversibility and directly impacts the computational stability of the calculations. Furthermore, the mass matrix in the global domain is directly influenced by grid coordinates and cell volumes, indicating that the quality of the computational grid significantly affects the calculation stability. Inadequate grid quality, especially anisotropic grids with greater skewness in boundary layers, can cause inaccurate inversion of the mass matrix and result in unstable calculations or even crashes.

Considering these aforementioned challenges, traditional DG methods encounter difficulties in achieving higher-order accuracy when simulating laminar and turbulent flows. Most simulations in global domains are typically limited to P2 (third-order), while P3 (fourth-order) accuracy can be achieved on simple grids such as equal-straight grids.

In this paper, a favorable aspect is the transformation of simulations into a reference domain, which effectively mitigates the influence of the grid on calculations and enhances the calculation accuracy to P3 or even P5, while ensuring computational robustness.

3.3 DG formulation in reference domain for RANS-SA equations

3.3.1 DG formulation in reference domain

In this work, calculations are conducted within the reference domain, which presents notable advantages over conventional global domain calculations in terms of stability and robustness when achieving high-order accuracy. Equation (13), when transformed into the reference domain, can be expressed as follows:

$$\begin{aligned}&\frac{\mathrm{d}}{\mathrm{d}t}\int_{\Omega_e}\mathbf{U}_h b_i \mathrm{d}\Omega+\int_{\Gamma_e}\mathbf{F}_k\left(\mathbf{U}_h^{\mathrm{L}},\mathbf{U}_h^{\mathrm{R}}\right)\mathbf{n}_k b_i \mathrm{d}\Gamma-\int_{\Omega_e}\mathbf{F}_k(\mathbf{U}_h)\frac{\partial b_i}{\partial\xi_j}\frac{\partial\xi_j}{\partial x_k}\mathrm{d}\Omega\\&=\int_{\Gamma_e}\mathbf{G}_k\left(\mathbf{U}_h^{\mathrm{L}},\nabla\mathbf{U}_h^{\mathrm{L}},\mathbf{U}_h^{\mathrm{R}},\nabla\mathbf{U}_h^{\mathrm{R}}\right)\mathbf{n}_k b_i \mathrm{d}\Gamma\\&\quad-\int_{\Omega_e}\mathbf{G}_k(\mathbf{U}_h)\frac{\partial b_i}{\partial\xi_j}\frac{\partial\xi_j}{\partial x_k}\mathrm{d}\Omega\\&\quad+\int_{\Omega_e}\mathbf{S}(\mathbf{U}_h,\nabla\mathbf{U}_h)b_i \mathrm{d}\Omega,\end{aligned}\tag{19}$$

where

$$\mathbf{U}_h=\sum_{i=1}^{N}\mathbf{U}_i(t)b_i(\xi),\tag{20}$$

with

$$\begin{aligned}&b_1=1,\ \ b_2=(\xi-\xi_c),\ \ b_3=(\eta-\eta_c),\ \ b_4=\frac{1}{2}(\xi-\xi_c)^2,\\&b_5=\frac{1}{2}(\eta-\eta)_c^2,\ \ b_6=(\xi-\xi_c)(\eta-\eta_c),\ \ b_7=\frac{1}{6}(\xi-\xi_c)^3,\\&b_8=\frac{1}{2}(\xi-\xi_c)^2(\eta-\eta_c),\ \ b_9=\frac{1}{2}(\xi-\xi_c)(\eta-\eta_c)^2,\\&b_{10}=\frac{1}{6}(\eta-\eta_c)^3,\ \ b_{11}=\frac{1}{24}(\xi-\xi_c)^4,\\&b_{12}=\frac{1}{6}(\xi-\xi_c)^3(\eta-\eta_c),\ \ b_{13}=\frac{1}{4}(\xi-\xi_c)^2(\eta-\eta_c)^2,\\&b_{14}=\frac{1}{6}(\xi-\xi_c)(\eta-\eta_c)^3,\ \ b_{15}=\frac{1}{24}(\eta-\eta_c)^4,\\&\cdots\end{aligned}\tag{21}$$

In the basis functions $b_i(\xi)$, ξ, and η represent the coordinates in the reference domain. The basis functions for calculating accuracy up to the 5th order are defined by Eq. (21), where the degrees of freedom of the basis functions increase with higher order accuracy calculations. Specifically, the degrees of freedom are 3, 6, 10, 15, and 21 for P1 to P5 accuracy, respectively.

3.3.2 *Mass matrix in reference domain*

In the reference domain, the semi-discrete form of Eq. (19) corresponds to Eq. (17). However, it should be noted that the form of the mass matrix differs across various computational domains. In the reference domain, the mass matrix can be denoted as follows:

$$\mathbf{M}=\int_{\Omega_e}b_i(\xi)b_j(\xi)\mathrm{d}\Omega.\tag{22}$$

The distinction between the mass matrices in Eqs. (18) and (22) lie in their respective basis functions. As indicated by the expression in Eq. (22), the mass matrix within the reference domain relies on the volume of the reference element, which presents a significant advantage by eliminating any complications associated with inverting the mass matrix due to the fixed shape and volume of the reference element. Therefore, transforming DG simulations into the reference domain can overcome the issue discussed in Sect. 3.2, where the stability of global domain calculations deteriorates due to the irreversibility of the mass matrix. This unique characteristic of our study enhances computational stability while achieving high-order accuracy.

3.3.3 *Roe scheme for the inviscid flux*

The convective flux of the Roe scheme [38] at DG framework in reference domain is defined as follows:

$$\begin{aligned}&\int_{\Gamma_e}\mathbf{F}_k\mathbf{n}_k b_i \mathrm{d}\Gamma-\int_{\Omega_e}\mathbf{F}_k(\mathbf{U}_h)\mathbf{n}_k\frac{\partial\xi_j}{\partial x_k}\frac{\partial b_i}{\partial\xi_j}\mathrm{d}\Omega\\&=\int_{\Gamma_e}\mathbf{F}_k\left(\mathbf{U}_h^{\mathrm{L}},\mathbf{U}_h^{\mathrm{R}}\right)\mathbf{n}_k b_i \mathrm{d}\Gamma-\int_{\Omega_e}\mathbf{F}_j^{\xi}(\mathbf{U}_h)\frac{\partial b_i}{\partial\xi_j}\mathrm{d}\Omega.\end{aligned}\tag{23}$$

The Roe flux is defined by

$$\mathbf{F}\left(\mathbf{U}_h^{\mathrm{L}},\mathbf{U}_h^{\mathrm{R}}\right)=\frac{1}{2}\left[\mathbf{F}\left(\mathbf{U}_h^{\mathrm{L}}\right)+\mathbf{F}\left(\mathbf{U}_h^{\mathrm{R}}\right)-\left|\widetilde{\mathbf{A}}\right|\left(\mathbf{U}_h^{\mathrm{R}}-\mathbf{U}_h^{\mathrm{L}}\right)\right],\tag{24}$$

where the related formulas of $\left|\widetilde{\mathbf{A}}\right|$ can be seen in Ref. [38].

3.3.4 *DDG method for the viscous flux*

The direct DG method, introduced by Liu and Yan [15], is based on the direct weak formulation for solving diffusion equations, and the viscous flux of DDG method is defined as follows:

$$\begin{aligned}&\int_{\Gamma_e}\mathbf{G}_k\mathbf{n}_k b_i \mathrm{d}\Gamma-\int_{\Omega_e}\mathbf{G}_k\frac{\partial\xi_j}{\partial x_k}\frac{\partial b_i}{\partial x_j}\mathrm{d}\Omega\\&=\int_{\Gamma_e}\mathbf{G}\left(\overline{\mathbf{U}},\nabla\overline{\mathbf{U}}\right)b_i \mathrm{d}\Gamma-\int_{\Omega_e}\mathbf{G}_k\left(\overline{\mathbf{U}},\nabla\overline{\mathbf{U}}\right)\frac{\partial\xi_j}{\partial x_k}\frac{\partial b_i}{\partial x_j}\mathrm{d}\Omega,\end{aligned}\tag{25}$$

where $\overline{\mathbf{U}}$ and $\nabla\overline{\mathbf{U}}$ are the cell-averaged values at the cell interface between the left and right element, which are defined as follows:

$$\begin{aligned}&\overline{\mathbf{U}}=\frac{1}{2}\left(\mathbf{U}_h^{\mathrm{L}}+\mathbf{U}_h^{\mathrm{R}}\right),\\&\overline{\mathbf{U}}_x=\beta_0\frac{\left(\mathbf{U}_h^{\mathrm{R}}-\mathbf{U}_h^{\mathrm{L}}\right)}{\Delta}n_x+\frac{1}{2}\left(\mathbf{U}_x^{\mathrm{R}}+\mathbf{U}_x^{\mathrm{L}}\right)\\&\qquad+\beta_1\Delta\left[\left(\mathbf{U}_{xx}^{\mathrm{R}}-\mathbf{U}_{xx}^{\mathrm{L}}\right)n_x+\left(\mathbf{U}_{xy}^{\mathrm{R}}-\mathbf{U}_{xy}^{\mathrm{L}}\right)n_y\right],\\&\overline{\mathbf{U}}_y=\beta_0\frac{\left(\mathbf{U}_h^{\mathrm{R}}-\mathbf{U}_h^{\mathrm{L}}\right)}{\Delta}n_y+\frac{1}{2}\left(\mathbf{U}_y^{\mathrm{R}}+\mathbf{U}_y^{\mathrm{L}}\right)\\&\qquad+\beta_1\Delta\left[\left(\mathbf{U}_{xy}^{\mathrm{R}}-\mathbf{U}_{xy}^{\mathrm{L}}\right)n_x+\left(\mathbf{U}_{yy}^{\mathrm{R}}-\mathbf{U}_{yy}^{\mathrm{L}}\right)n_y\right],\end{aligned}\tag{26}$$

and the derivatives in reference domain are written as

$$\begin{aligned}&\mathbf{U}_x=\mathbf{U}_\xi\xi_x+\mathbf{U}_\eta\eta_x,\\&\mathbf{U}_y=\mathbf{U}_\xi\xi_y+\mathbf{U}_\eta\eta_y,\\&\mathbf{U}_{xx}=\mathbf{U}_{\xi\xi}\xi_x^2+2\mathbf{U}_{\xi\eta}\xi_x\eta_x+\mathbf{U}_{\eta\eta}\eta_x^2+O\left(\xi_{x_\xi},\eta_{x_\xi}\right),\\&\mathbf{U}_{yy}=\mathbf{U}_{\xi\xi}\xi_y^2+2\mathbf{U}_{\xi\eta}\xi_y\eta_y+\mathbf{U}_{\eta\eta}\eta_y^2+O\left(\xi_{y_\xi},\eta_{y_\xi}\right),\\&\mathbf{U}_{xy}=\mathbf{U}_{\xi\xi}\xi_x^2+\mathbf{U}_{\xi\eta}\left(\xi_x\eta_y+\xi_y\eta_x\right)+\mathbf{U}_{\eta\eta}\eta_x^2+O\left(\xi_{x_\xi},\eta_{x_\xi}\right),\end{aligned}\tag{27}$$

where β_0 and β_1 are coefficients, and Δ is computed by Ref. [33],

$$\Delta = \Delta_L + \Delta_R, \tag{28}$$

where Δ_L and Δ_R denote the distances of the centroid to the common face from the left and right elements, respectively. For the DDGP1 and DDGP2 schemes, the coefficients β_0 range from 2.0 to 6.0 [39], while β_1 is equal to 1/12. For higher order accuracy calculations, it is recommended to use a larger value of β_0. Test cases presented in this paper demonstrate that setting β_0 equals to 8.0-20.0 yields satisfactory results for higher order accuracy calculations within the reference domain.

4. Implementation of implicit robust DG solver

The spatial discretization of the RANS-SA system, as described in Eq. (17), can be linearized over time and expressed as follows:

$$\mathbf{A}\Delta\mathbf{U} \cong -\mathbf{R}^n, \quad \mathbf{A} = \frac{\mathbf{M}}{\Delta t}\mathbf{I} + \frac{\partial \mathbf{R}^n}{\partial \mathbf{U}}, \tag{29}$$

Where $\partial\mathbf{R}^n/\partial\mathbf{U}$ represents the Jacobian matrix, Δt denotes the time increment, and $\Delta\mathbf{U} = \mathbf{U}^{n+1} - \mathbf{U}^n$ refers to the solution difference between time levels n and $n + 1$. It should be noted that the robustness of the solver for the system described in Eq. (29), particularly in the case of high-order accuracy solutions, is heavily influenced by the accuracy and stiffness of matrix **A**. The accuracy of matrix **A**, in terms of its structure, is primarily determined by the mass matrix **M** and Jacobian matrix $\partial\mathbf{R}^n/\partial\mathbf{U}$, which serve as crucial factors significantly impacting computational stability.

4.1 GMRES method

The GMRES method has emerged as a fundamental technique in the field of numerical analysis, particularly in the context of resolving nonsymmetric linear systems stemming from partial differential equations. One of the pivotal merits attributed to the GMRES method lies in its optimality characteristic, guaranteeing convergence towards a solution within the minimal number of iterations conceivable. Moreover, it exhibits exceptional performance in practical scenarios, yielding accurate results even amidst the presence of highly nonlinear or ill-conditioned systems. Due to its efficiency and dependability, the GMRES method has attained widespread adoption in CFD.

In this paper, GMRES method with a preconditioner is chosen to solve the linear system of Eq. (29). The preconditioning technique stands as a typical strategy employed to enhance the convergence behavior of the GMRES method. Equation (29), featuring a left preconditioning matrix denoted as **P**, can be formulated as follows:

$$\mathbf{P}^{-1}\mathbf{A}\Delta\mathbf{U} = \mathbf{P}^{-1}\mathbf{R}^n. \tag{30}$$

The pseudo-code for the solution procedure is presented as Algorithm 1.

Algorithm 1 GMRES method with preconditioner

1: $n = 0$, Itermax = kxgmres, $\Delta\mathbf{U} = \mathbf{0}$
2: **for** $n <$ ngmres **do**
3: $r_0 = \mathbf{R}^n - \mathbf{A}\Delta\mathbf{U}$, $\beta = \|r_0\|$, $v_1 = r_0/\beta$
4: **if** $n = 0$ **then**
5: $\beta_0 = \beta$
6: **end if**
7: **for** $j = 1, 2, ...,$ Itermax **do**
8: $w_j = \mathbf{A}v_j$, $z_j = \mathbf{P}^{-1}w_j$
9: **for** $i = 1, 2, ..., j$ **do**
10: $h_{i,j} = (z_j \cdot v_i)$, $z_j = z_j - h_{i,j}v_i$
11: **end for**
12: $h_{j+1,j} = \|z_j\|$, $v_{j+1} = z_j/h_{j+1,j}$
13: **end for**
14: Using QR decomposition method to get the new $\Delta\mathbf{U}'$
15: $r' = \mathbf{R}^n - \mathbf{A}\Delta\mathbf{U}'$, $\beta' = \|r'\|$.
16: res $= \beta'/\beta_0$
17: **if** res < GMRES_res **then**
18: exit GMRES sub iteration
19: **else**
20: $n = n + 1$
21: Itermax = Itermax + $n \cdot$ kygmres
22: **end if**
23: **end for**

In the pseudocode, kxgmres represents the initial Krylov subspace, kygmres denotes the increment of the Krylov subspace at each subsequent GMRES sub-iteration, ngmres specifies the maximum number of sub-iterations, and controls the upper limit on the size of the Krylov subspace. GMRES_res refers to the convergence criterion for GMRES. Appropriately configuring these parameters can enhance the convergence and stability of the calculations, whereas improper settings may lead to over-solutions [34,35] or divergence. In Sect. 5, the impact of these parameters on the calculations will be comprehensively analyzed through test cases, with a particular emphasis on their effects on high-order DG calculations.

References [27,36] have demonstrated that the performance of GMRES is significantly influenced by the accuracy of preconditioning and matrix-vector generation in line 8 of the pseudocode, which are closely tied to the Jacobian matrix. These factors play a critical role in determining the robustness and efficiency of the computations. Generally, preconditioning can be approximated using an approximate Jacobian matrix, while an exact calculation of matrix **A** is required for matrix-vector generation. The conventional treatment approach typically employs an inviscid flux Jacobian with a spectral radius instead of a viscous flux Jacobian matrix for preconditioning, and finite difference

calculation is utilized for matrix-vector generation. Evidently, this necessitates computing the Jacobian matrix twice.

In this paper, the LU-SGS method is employed for preconditioning, and the **L**, **U**, and **D** matrices are computed using a strategy that solves for an exact Jacobian matrix. The advantage of employing this approach is that the resulting **L**, **U**, and **D** matrices can be directly used for matrix-vector generation, thereby obviating the need for finite difference calculations and eliminating any additional computational or storage requirements. Moreover, the LU-SGS preconditioner utilizing the exact Jacobian matrix reduces the stiffness of the system and significantly improves both the stability and convergence of the GMRES method [40,41].

4.2 Exact Jacobian matrix

For RANS equations coupled with the negative-SA turbulence model, the Jacobian of DG discretization comprises three components [33]: the inviscid flux Jacobian, the viscous flux Jacobian, and the source term Jacobian. These components can be derived as follows:

$$\mathbf{A}_{\text{inv}} = \begin{cases} \mathbf{L}_{\text{inv}} = \sum\limits_{I<J}^{nf} \dfrac{\partial \mathbf{F}}{\partial \mathbf{U}^I} b_i{}^J \Gamma, \\ \mathbf{D}_{\text{inv}} = \sum\limits_{I=J}^{nf} \dfrac{\partial \mathbf{F}}{\partial \mathbf{U}^I} b_i{}^I \Gamma - \dfrac{\partial \mathbf{F}_k}{\partial \mathbf{U}} \dfrac{\partial \xi_j}{\partial x_k} \dfrac{\partial b_i}{\partial \xi_j} \Omega, \\ \mathbf{U}_{\text{inv}} = \sum\limits_{I>J}^{nf} \dfrac{\partial \mathbf{F}}{\partial \mathbf{U}^I} b_i{}^J \Gamma, \end{cases} \tag{31}$$

$$\mathbf{A}_{\text{vis}} = \begin{cases} \mathbf{L}_{\text{vis}} = -\sum\limits_{I<J}^{nf} \dfrac{\partial \mathbf{G}}{\partial \mathbf{U}^I} b_i{}^J \Gamma, \\ \mathbf{D}_{\text{vis}} = -\sum\limits_{I=J}^{nf} \dfrac{\partial \mathbf{G}}{\partial \mathbf{U}^I} b_i{}^J \Gamma + \dfrac{\partial \mathbf{G}_k}{\partial \mathbf{U}} \dfrac{\partial \xi_j}{\partial x_k} \dfrac{\partial b_i}{\partial \xi_j} \Omega - \dfrac{\partial \mathbf{S}}{\partial \mathbf{U}} b_i \Omega + \dfrac{\mathbf{M}}{\Delta t} \mathbf{I}, \\ \mathbf{U}_{\text{vis}} = -\sum\limits_{I>J}^{nf} \dfrac{\partial \mathbf{G}}{\partial \mathbf{U}^I} b_i{}^J \Gamma, \end{cases} \tag{32}$$

where *nf* represents the total number of faces in triangle or quadrilateral elements, and the larger value between *I* and *J* denotes the right (R) state of the element interface while the other represents its left (L) state. Although both the inviscid and viscous flux formulations are intricate, the DG method's compactness enables expressing the derivatives of these fluxes as functions of the test function. Therefore, the derivatives can be computed using the chain rule.

References [32,33] have demonstrated the advantages of employing an exact Jacobian matrix in low-order calculations. It should be noted that different schemes yield different forms of Jacobians. In this paper, we utilize Roe's approximated Riemann solver for the inviscid flux and the DDG scheme for the viscous flux. The viscous flux Jacobian of the DDG scheme and the source term Jacobian of the negative SA turbulence model have been detailed in Ref. [33], while we will provide a comprehensive derivation of the inviscid Jacobian in Appendix A.

4.3 Adaptive CFL number

The CFL number plays a significant role in determining the computational efficiency [42-44], with higher values generally leading to greater efficiency. However, it is important to note that if the CFL number becomes excessively large, it can cause instability and divergence. Therefore, striking the right balance between stability and efficiency when working with implicit schemes can be quite challenging. To overcome this issue, the concept of an adaptive CFL number emerges as a potential solution. By employing an adaptive approach that dynamically adjusts the CFL number in response to changes in the flow field, a better trade-off between stability and efficiency can be achieved during the simulation.

The adaptive CFL number used in this paper is given as follows:

$$CFL^{n+1} = \begin{cases} CFL^n (CFL_{\max} - CFL_{\text{init}})^{1/(CFL_{\text{step}}-1)}, & \text{if } FCFL \le 0.1, \\ CFL^n, & \text{if } FCFL > 0.1, \end{cases} \tag{33}$$

where CFL^{n+1} and CFL^n denote the CFL number between time level n and $n+1$. In ideal state, the solution process is steady and the condition $FCFL > 0.1$ will not be triggered, then, the CFL number will increase from CFL_{init} to $CFL_{\max}$ within CFL_{step} steps. FCFL is a quantity related to the change of flow field variables, and computed from

$$FCFL = \frac{\|\Delta \mathbf{U}\|}{\|\mathbf{U}\|}, \tag{34}$$

where $\|\Delta \mathbf{U}\|$ represents the L_2 norm of the increment of conservative variables vector, and $\|\mathbf{U}\|$ denotes the L_2 norm of the conservative variables vector in the time level n. In physical terms, this ratio describes changes in physical quantities within a flow field. The purpose of imposing a limiter on the FCFL is to mitigate excessive variations in physical quantities resulting from abrupt increases in the CFL number. When the aforementioned ratio exceeds a value significantly greater than 10, the computations are susceptible to collapse, necessitating a reduction in the growth rate of the CFL number during iterations. The aforementioned CFL increasing model exhibits a time delay, although our test cases demonstrate satisfactory performance when a limiter of FCFL is set to 0.1. Furthermore, reducing the FCFL limiter or increasing the CFL_{step} can decelerate the growth rate of the CFL number for complex flow simulations.

5. Numerical results

To evaluate the proposed method in this paper, several representative cases have been selected. These cases aim to showcase the performance of implicit high-order calculations and analyze the influential parameters. The selected cases include one case of laminar steady flow, three cases of turbulent steady flow.

5.1 Laminar flow past a circular cylinder

In this test case, the laminar flow past a circular cylinder is considered. The initial condition is a uniform flow where the Mach number equals to 0.2 and the Reynolds number equals to 40 based on the diameter of the cylinder. The computational domain employs a hybrid mesh comprising 8406 triangular elements, 4568 quadrilateral elements, and 100 points on the cylinder's surface, as depicted in Fig. 1. The domain boundaries span from −25.0 to 60.0 in the x-direction and −25.0 to 25.0 in the y-direction, with the cylinder centered at (0,0). The cell size at the cylinder is $0.0337d$, where d represents the diameter of the cylinder.

Firstly, a comparison is conducted between the computational efficiency of LU-SGS and GMRES. Figure 2 presents the convergence histories of both methods under an equal number of CFL iterations. It is evident that due to sub-iterations within GMRES, its single-step CPU time surpasses that of LU-SGS. However, the computational results demonstrate that regardless of whether the DDGP1 or DDGP2 scheme is utilized, LU-SGS requires more iteration steps and CPU time than GMRES to achieve convergence. For instance, considering DDGP2, the number of iteration steps (12813) and iteration CPU time (15690.60 s) for LU-SGS are 15.5 times (824) and 3.45 times (4541.26 s) that of GMRES, respectively. Thus, the calculation efficiency of GMRES surpasses that of LU-SGS.

Secondly, the influence of adaptive CFL numbers on the computational efficiency of GMRES is discussed. The calculations are performed using a fixed CFL number of 100, as well as increasing CFL numbers ranging from 100 to 10000 and 1000000 within 100 iterations. Convergence histories for the DDGP1 and DDGP2 schemes under vary-

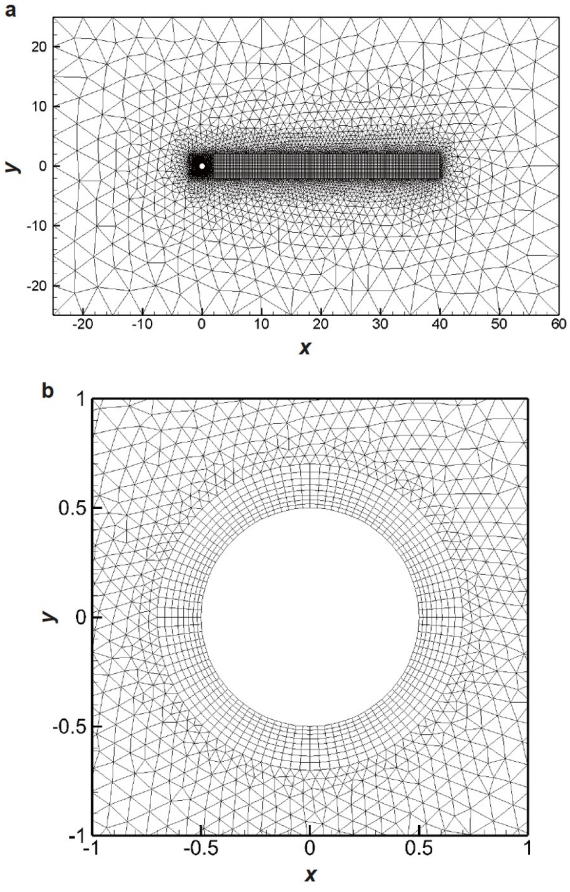

Figure 1 Hybrid mesh for a laminar flow past a cylinder: **a** computational domain; **b** zoom view.

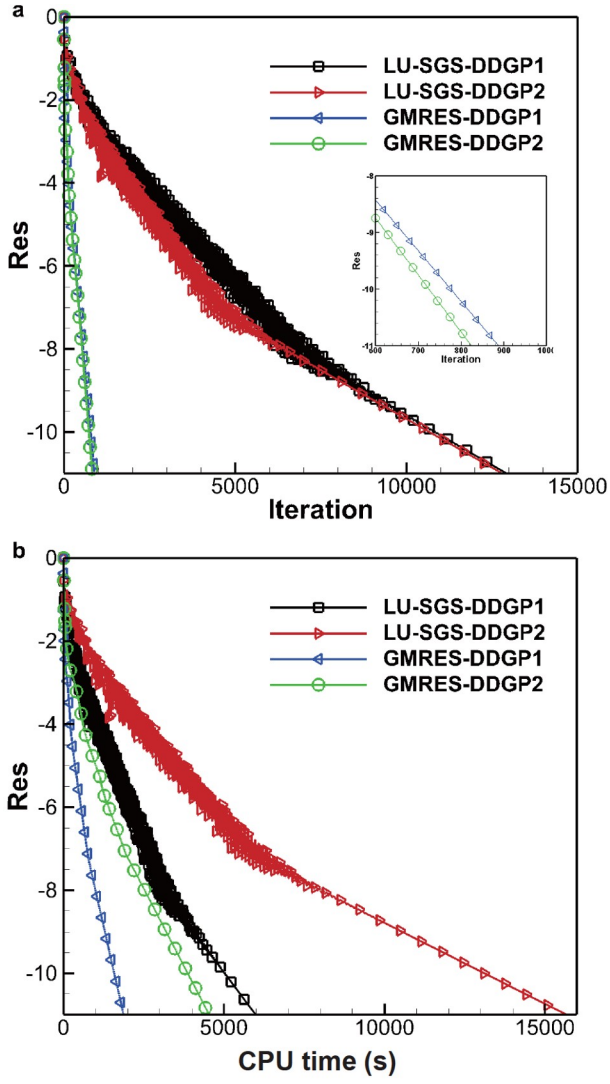

Figure 2 Convergence histories of LU-SGS and GMRES: **a** residual-iteration; **b** residual-CPU time (s).

ing strategies of CFL number increase are presented in Figs. 3 and 4. It becomes evident that the adaptive CFL number accelerates the convergence rate while ensuring computational stability, in comparison to a fixed CFL number.

Finally, the GMRES method, in conjunction with an adaptive CFL number, is employed for higher-order calculations. The GMRES method is initialized with a Krylov subspace consisting of 10 vectors, and the subspace size is increased by 10 at each sub-iteration until convergence is achieved at 0.01 or a maximum subspace size of 100 is reached. Figure 5 presents the convergence histories of the DDGP3-P5 schemes, illustrating that the proposed method facilitates stable and rapid convergence to the specified convergence standard. The computational time required for the convergence of DDGP1-P5 calculations is presented in Table 1. It can be observed that as the accuracy improves, the computational time increases. Under the same CFL number growth strategy, the DDGP5 scheme necessitates approximately 107.2 times more computational time compared to the DDGP1 scheme. This discrepancy can be attributed not only to the significantly higher degrees of freedom in the DDGP5 scheme but also to GMRES's tendency towards over-solving.

Figure 6 showcases the computed density contours and streamlines over the cylinder obtained using DDGP1 to DDGP5 schemes. A notable characteristic of this test case is the occurrence of flow separation after the cylinder. Figure 7 illustrates the convergence history of the total drag coefficient as the order of DDG schemes increases. It clearly shows that as the order increases, the total drag coefficient converges to a stable value, and the error between the calculated value and the reference value diminishes. Table 2 provides numerical values for the form drag, skin friction drag, total drag coefficients, and the length of the recirculation bubble. A comparison with the reference value [45] reveals that the solutions obtained using DDG schemes are acceptable, with all drag coefficient errors remaining

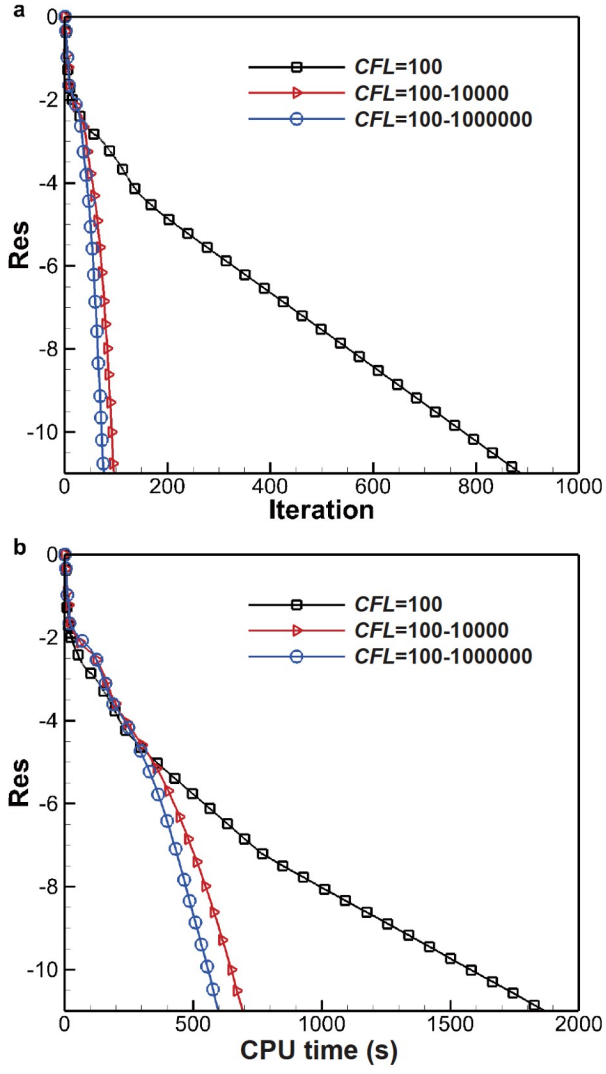

Figure 3 Influence of CFL number to the convergence of GMRES (DDGP1 scheme): **a** residual-iteration; **b** residual-CPU time (s).

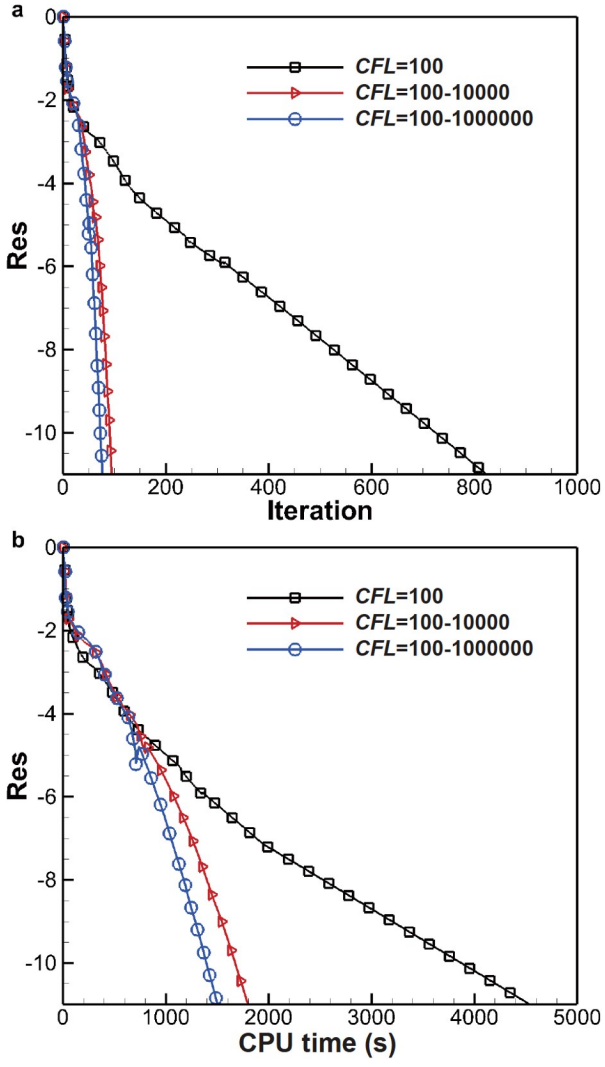

Figure 4 Influence of CFL number to the convergence of GMRES (DDGP2 scheme): **a** residual-iteration; **b** residual-CPU time (s).

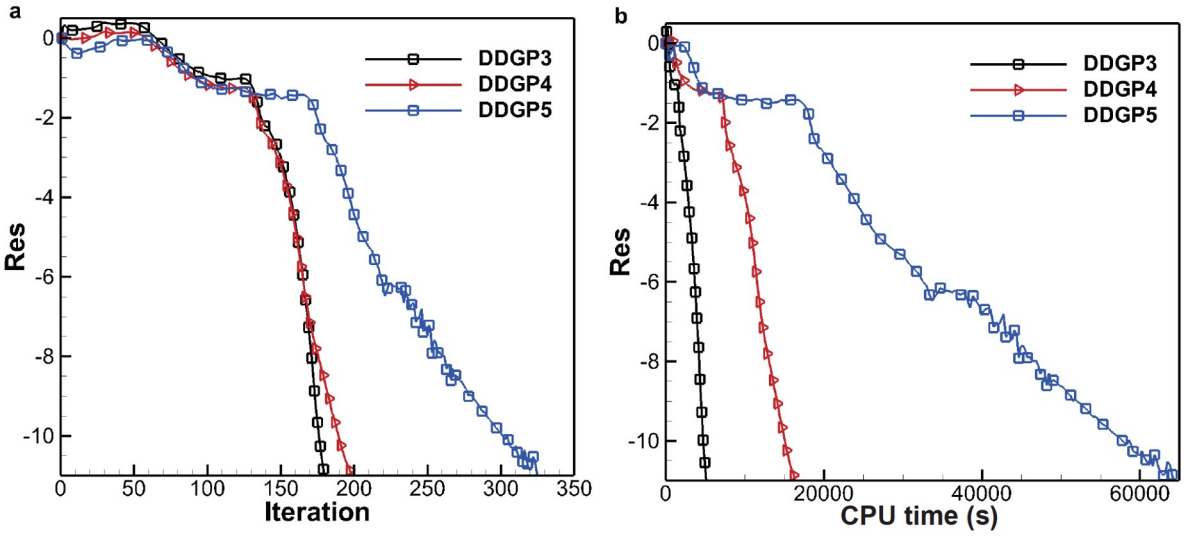

Figure 5 Convergence histories of DDGP3-P5 schemes: **a** residual-iteration; **b** residual-CPU time (s).

Table 1 CPU times of DDGP1-P5 schemes under different CFL number increasing strategy

Scheme	CPU time (s)		
	Fixed CFL number	Adaptive CFL number	
	CFL = 100	*CFL* = 100-10^4	*CFL* = 100-10^6
DDGP1	1868.96	692.68	599.90
DDGP2	4541.26	1822.23	1516.12
DDGP3	–	–	5148.07
DDGP4	–	–	16489.20
DDGP5	–	–	64337.71

below 1.0%. Moreover, DDGP3 to DDGP5 schemes yield nearly identical results. The numerical results indicate that the proposed method can achieve satisfactory calculation accuracy for separation flow.

5.2 Turbulent flow past a NACA0012 airfoil

In this particular test case, a turbulent flow past a NACA0012 airfoil is considered. This test case is a standard validation case. The flow parameters include a Mach number of 0.15, Reynolds number of 6.0×10^6, and an attack angle of 10°. The test grid employed consists of a hybrid mesh comprising 4637 triangular elements and 6702 quadrilateral elements, as illustrated in Fig. 8. The approximate average $y+$ value is approximately 3.0, with the first layers of the mesh around the wall boundary having an average spacing of 1.0×10^{-5}. The far-field boundary in the provided mesh is located approximately 200 chord lengths away from the airfoil.

This calculation studies the impact of Krylov subspace growth mode on the computation. Table 3 presents the settings for each Krylov subspace growth mode. Initially, calculations based on DDGP1 and DDGP2 schemes are performed using a fixed Krylov subspace (F_Krylov subspace) and an incremental Krylov subspace (I_Krylov subspace). The convergence histories of these two calculations are illustrated in Fig. 9 . It is evident that while the F_Krylov subspace calculation requires less computational time, its sub-iteration within GMRES fails to meet the convergence criteria, leading to significant instability during computation. In contrast, it can be observed from the convergence history that utilizing an I_Krylov subspace results in a more stable computational process with fewer iterations required for convergence. The instability of the convergence process in F_Krylov subspaces may lead to divergence or non-convergence of higher-order calculations, as confirmed by subsequent computations. Figures 10-12 depict the computational convergence from the DDGP3 to the DDGP5 scheme. In F_Krylov subspace calculations, the residuals fail to decrease, whereas I_Krylov subspace calculations exhibit better convergence.

Figure 13 gives the convergence history of lift and drag coefficients with the order of DDG schemes and Table 4 summarizes the specific value of total drag coefficients, total lift coefficients, form drag, and skin friction using the I_Krylov subspace. The results demonstrate that the total lift and drag coefficients obtained using DDG schemes are all acceptable and closely approximate the numerical results calculated by CFL3D [46] using a finer grid (229376 elements). As the accuracy order increases, the calculated re-

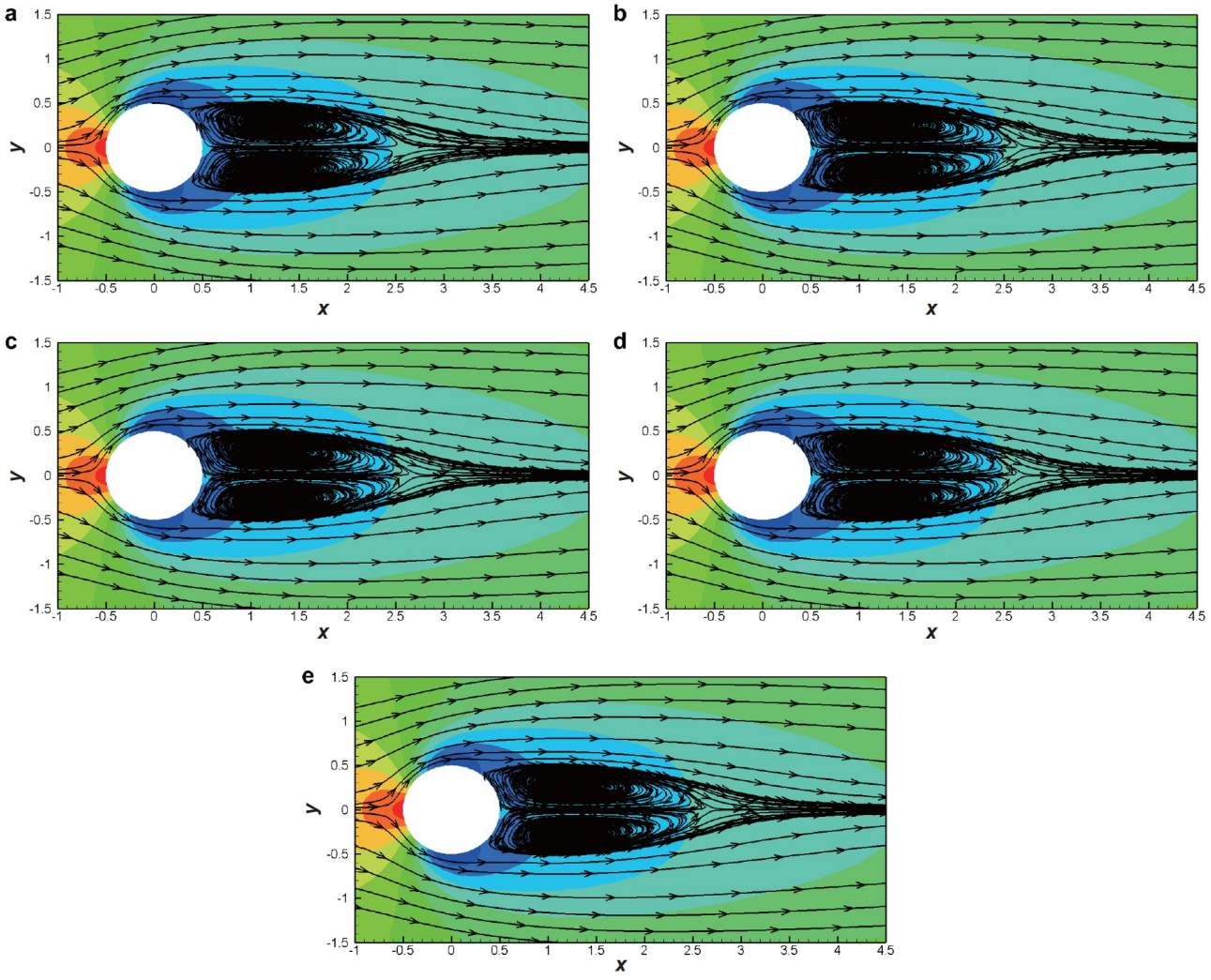

Figure 6 Density contours and streamlines of DDG schemes: **a** DDGP1; **b** DDGP2; **c** DDGP3; **d** DDGP4; **e** DDGP5.

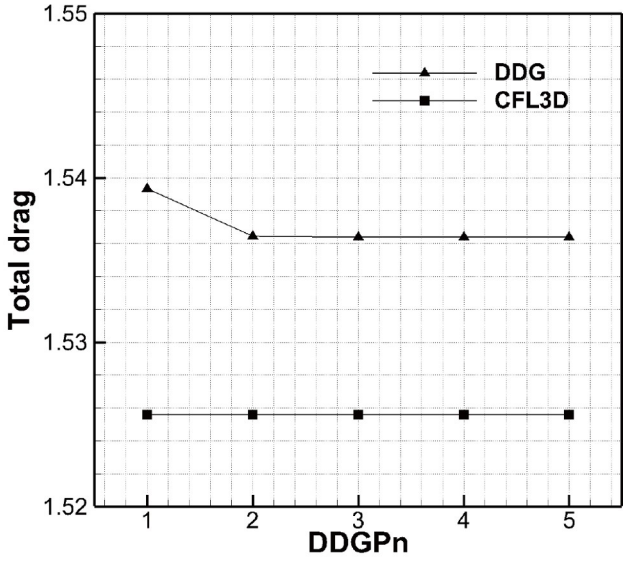

Figure 7 Convergence history of total drag coefficient with the order of DDG schemes.

sults approach the reference value, with minimum drag coefficient and lift coefficient errors of 0.39% and 1.62%, respectively. It is worth noting that the grid density in Ref. [46] is approximately 11.7 times greater than that employed in this test case. The enhancement in order accuracy of DDG schemes compensates for the loss of calculation precision arising from insufficient mesh density. Figure 14 gives the Mach contours and streamline obtained by DDGP5 scheme.

5.3 Turbulent flow past a NACA4412 airfoil

In this test case, a turbulent flow past a NACA4412 airfoil is considered. The relevant flow parameters include a Mach number of 0.09, a Reynolds number of 1.52×10^6 based on the free-stream velocity and the chord length of the airfoil, and an angle of attack (AOA) of 13.87°. The computational analysis is carried out employing a hybrid mesh configuration, as illustrated in Fig. 15, comprising 4964 triangular elements and 6383 quadrilateral elements. The mesh design ensures an approximate average $y+$ value below 1.0, with

Table 2 Drag coefficients and the recirculation length obtained by DDG schemes

Scheme	Form drag	Skin friction	Total drag	Recirculation length
DDGP1	1.01372	0.52562	1.53934	2.14
DDGP2	1.01249	0.52396	1.53645	2.29
DDGP3	1.01255	0.52383	1.53638	2.28
DDGP4	1.01255	0.52383	1.53638	2.28
DDGP5	1.01255	0.52383	1.53638	2.28
Ref. [45]	1.0260	0.4993	1.5256	2.27

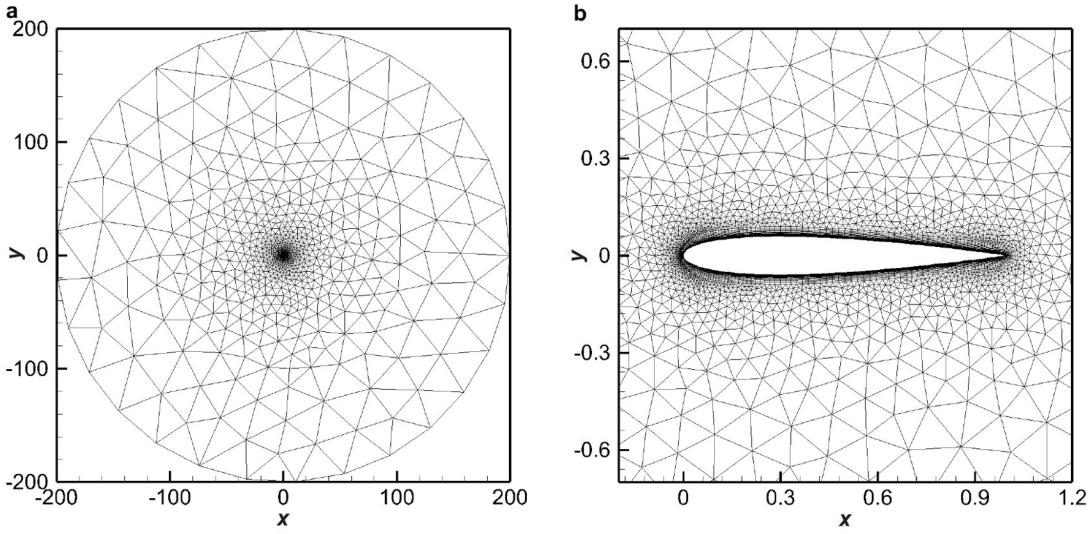

Figure 8 Hybrid mesh for turbulent flow over a NACA0012 airfoil at attack angle of 10°: **a** computational domain; **b** zoom view.

Table 3 Settings for Krylov subspace growth mode

Krylov subspace growth mode	Kxgmres	Kygmres	Ngmres	GMRES_res
F_Krylov subspace	30	0	0	–
I_Krylov subspace	30	10	7	0.01

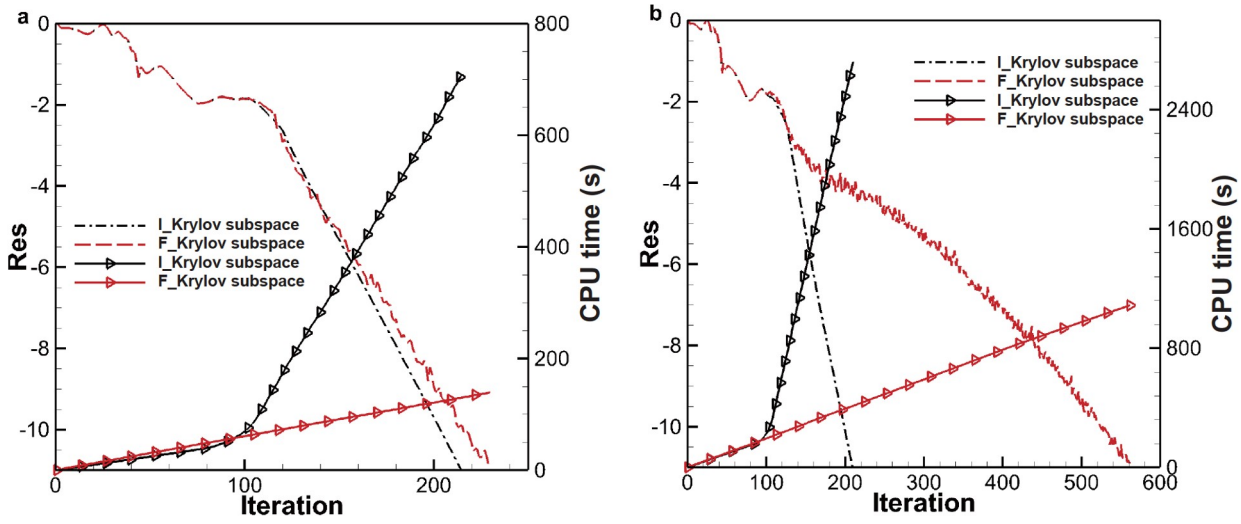

Figure 9 Influence of Krylov subspace to the convergence of GMRES: **a** DDGP1; **b** DDGP2.

the first layers around the wall boundary exhibiting an average spacing of 1.0×10^{-5}. It is worth noting that the far field boundary within the provided mesh is situated nearly 100 chord lengths away from the airfoil.

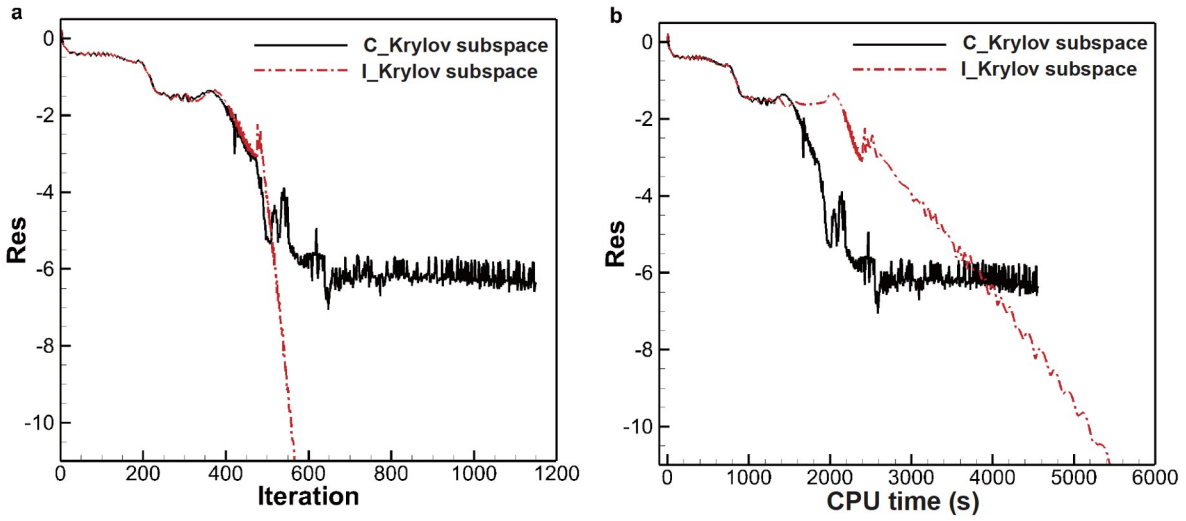

Figure 10 Influence of Krylov subspace to the convergence of GMRES (DDGP3): **a** residual-iteration; **b** residual-CPU time (s).

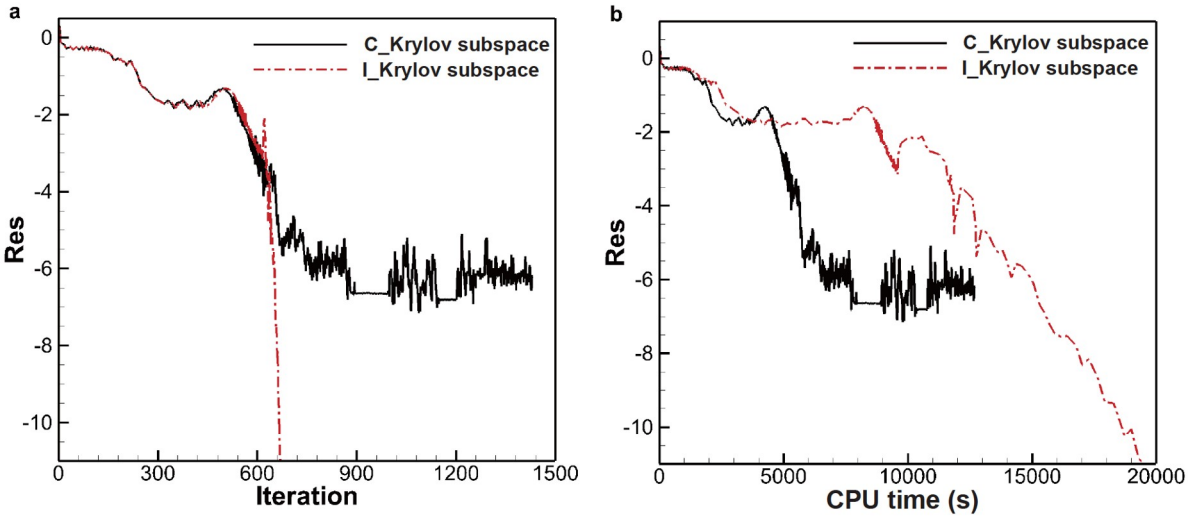

Figure 11 Influence of Krylov subspace to the convergence of GMRES (DDGP4): **a** residual-iteration; **b** residual-CPU time (s).

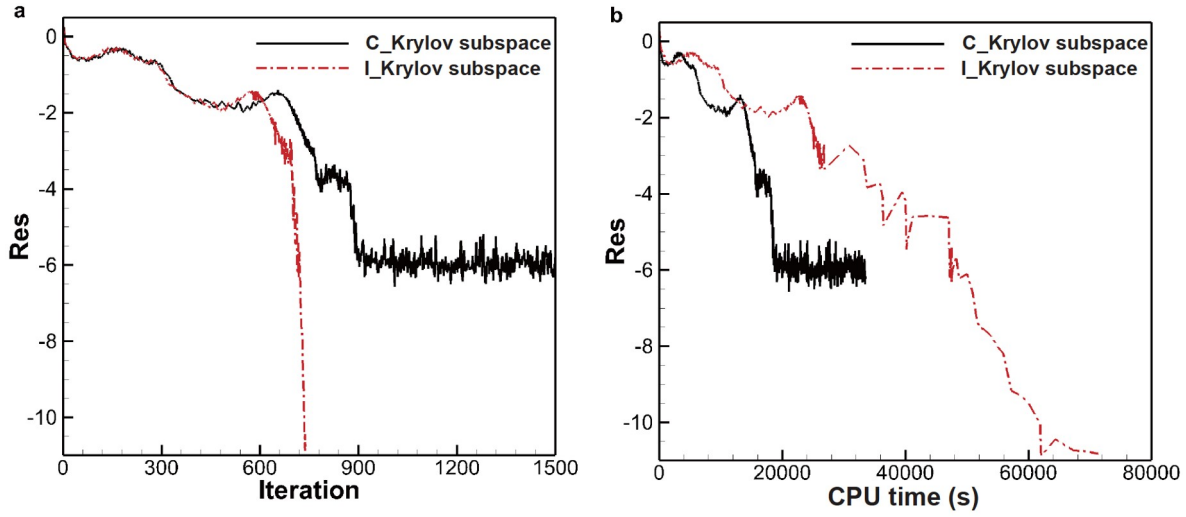

Figure 12 Influence of Krylov subspace to the convergence of GMRES (DDGP5): **a** residual-iteration; **b** residual-CPU time (s).

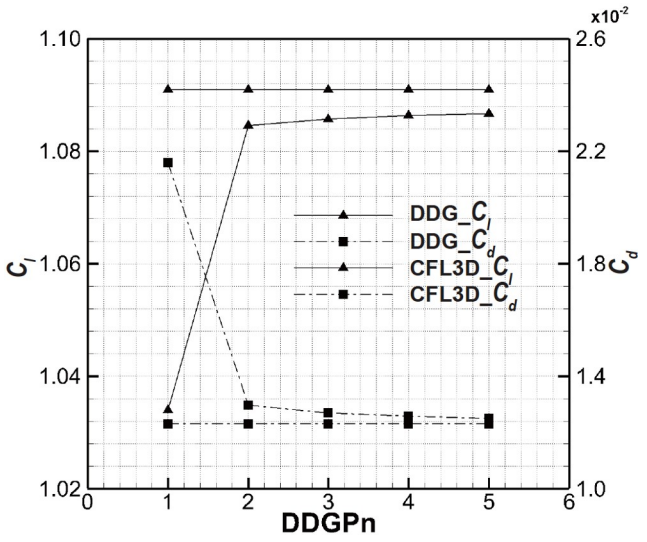

Figure 13 Convergence history of total lift and drag coefficients with the order of DDG schemes.

It is widely acknowledged that achieving convergence for turbulent flow past an airfoil at high angles of attack presents considerable challenges due to the occurrence of flow separation near the trailing edge, which leads to the formation of a large recirculation bubble in the wake region. Nevertheless, employing the methodology expounded in this study, it becomes feasible to attain rapid convergence of DDGP1 to DDGP4 by increasing the CFL number from 1 to 1.0×10^6 within 500 steps, setting the residual convergence criteria of GMRES to 0.001, and employing the I_Krylov subspace mode. As portrayed in Fig. 16, the computation residuals swiftly converge to 11 orders of magnitude, despite the presence of a notable recirculation bubble near the trailing edge, which is distinctly observable in Fig. 17. Another adverse result of flow separation lies in the intricate prediction of lift and drag coefficients, with the latter proving particularly challenging. Figure 18 gives the convergence history of total lift and drag coefficient with the order of DDG schemes and the specific value of these coefficients, form drag, and skin friction are succinctly summarized in Table 5. A comparison between the computed values and those obtained from CFL3D [46] reveals that, as the calculation order improves, the computational error reduces. This case highlights the benefits of order enhancement in computational accuracy, and it should be noted that the grid density in the literature is approximately 23.5 times greater than that employed in this test case. More specifically, the lift and drag coefficients calculated using the DDGP4 scheme exhibit errors of 0.87% and 4.50%, respectively. Specifically, the pressure coefficients acquired through the DDGP4 scheme demonstrate satisfactory computational accuracy, as illustrated in Fig. 19.

In addition, this test case also explores the impact of

Table 4 Comparison of lift and drag coefficients of DDG schemes

$AOA = 10°$	C_l	C_d	C_{dp}	C_{dv}
DDGP1	1.03398	0.02159	0.01645	0.00514
DDGP2	1.08453	0.01297	0.00707	0.00590
DDGP3	1.08566	0.01270	0.00665	0.00605
DDGP4	1.08634	0.01259	0.00647	0.00612
DDGP5	1.08667	0.01251	0.00638	0.00613
CFL3D [46]	1.0909	0.01231	□	□

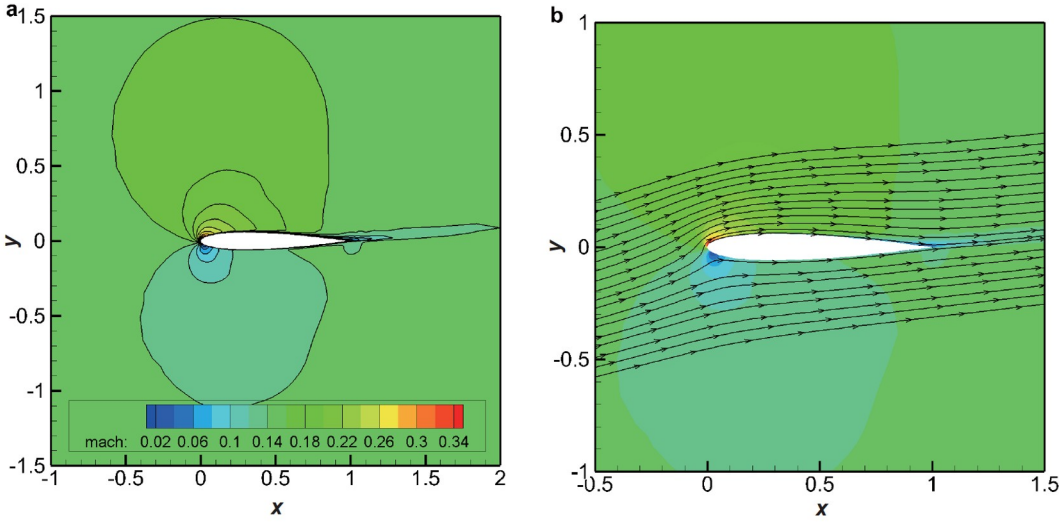

Figure 14 Mach contours and streamline obtained by DDGP5 scheme: **a** mach contours; **b** streamline.

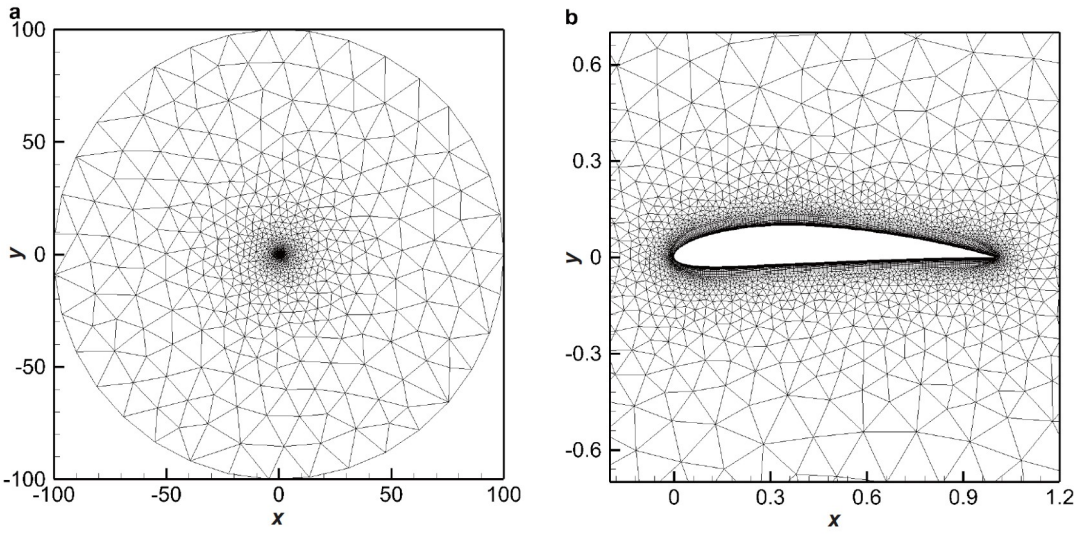

Figure 15 Hybrid mesh for turbulent flow over a NACA4412 airfoil at attack angle of 13.87°: **a** computational domain; **b** zoom view.

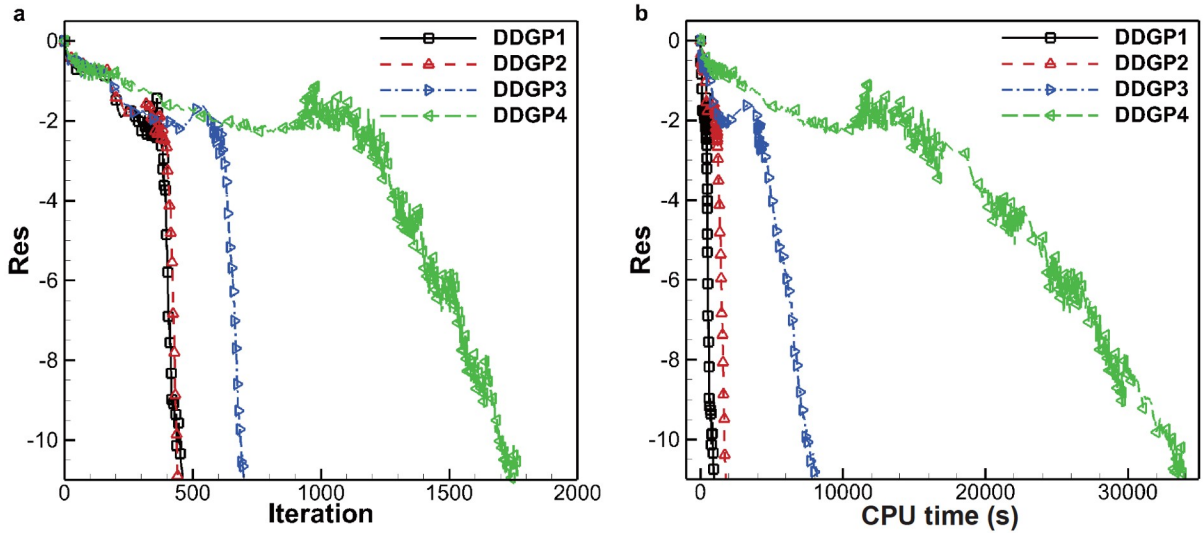

Figure 16 Comparison of convergence histories of DDGP1 to DDGP4 schemes: **a** residual-iteration; **b** residual-CPU time (s).

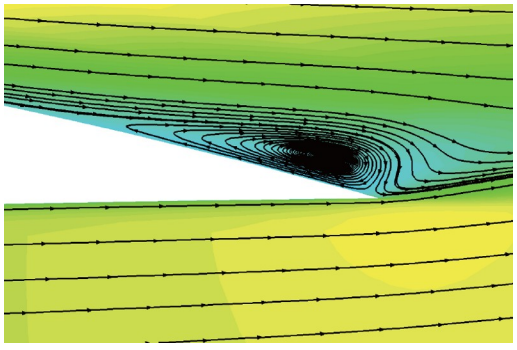

Figure 17 Streamlines for turbulent flow past a NACA4412 airfoil at attack angle of 13.87°.

GMRES convergence tolerance on calculation convergence. All test cases involve an increase in the CFL number from 1 to 1.0×10^6 within 500 steps. Figure 20 presents the computational convergence histories and CPU time of calculations based on different order DDG schemes. It is notable from the computational results obtained through DDGP1 and DDGP2 schemes that excessively small GMRES convergence tolerance during low-order calculations may result in over-solution, leading to prolonged single-step GMRES sub-iterations. Conversely, the computations demonstrate that larger GMRES convergence tolerances are capable of satisfying the convergence requirements of low-order calculations. The calculation results derived from DDGP3 and DDGP4 schemes indicate that for higher-order calculations, the GMRES convergence tolerance should be reduced; otherwise, the calculation may fail to converge or even diverge. Remarkably, the results obtained through the DDGP4 scheme reveal that, although a convergent solution can be

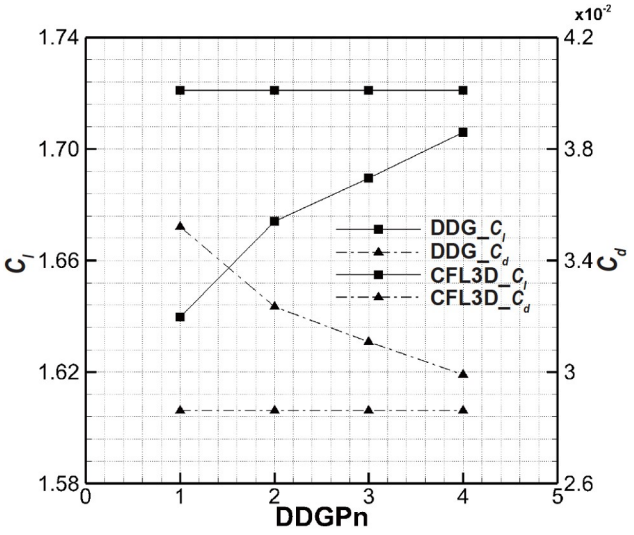

Figure 18 Convergence history of total lift and drag coefficient with the order of DDG schemes.

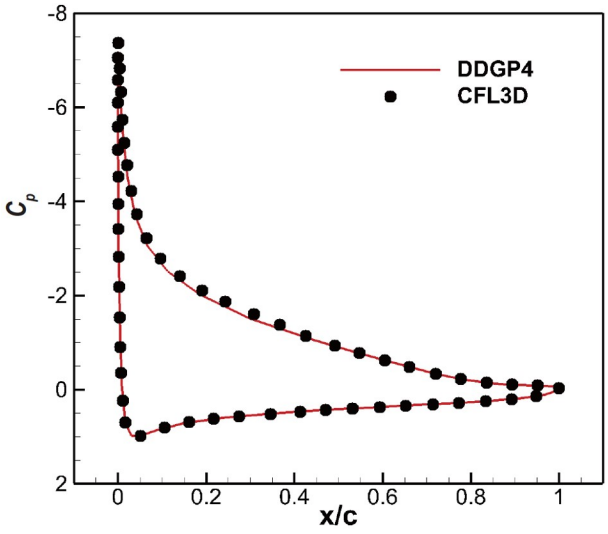

Figure 19 Pressure coefficients of DDG methods for turbulent flow past a NACA4412 airfoil.

Table 5 Comparisons of lift and drag coefficients of DDG schemes

$AOA = 13.87°$	C_l	C_d	C_{dp}	C_{dv}
DDGP1	1.63970	0.03521	0.03005	0.00516
DDGP2	1.67396	0.03234	0.02582	0.00652
DDGP3	1.68950	0.03109	0.02441	0.00668
DDGP4	1.70596	0.02990	0.02312	0.00678
CFL3D [46]	1.7210	0.02861	0.02156	0.00704

obtained with higher GMRES convergence tolerance, the convergence process is more volatile when compared to utilizing lower GMRES convergence tolerance. Moreover, it fails to fully capture the rapid convergence characteristic of the GMRES method.

5.4 Turbulent flow past a 30P30N multi-element airfoil

In this test case, turbulent flow past a 30P30N multi-element airfoil is also studied. The initial condition corresponds to a uniform flow featuring a Mach number of 0.17, a Reynolds number of 1.7×10^6 based on the free-stream velocity and the chord length of the airfoil, and an AOA of 5.5°. The simulations are conducted utilizing a hybrid mesh, as depicted in Fig. 21, which consists of 1033 triangular and 14139 quadrilateral elements. The first layers of the mesh in proximity to the wall boundary are characterized by an average spacing of 2.0×10^{-5}, while the far field boundary is situated at a distance of 200 chord lengths away from the airfoil.

For the simulation of turbulent flow with complex geometry, achieving convergence poses a formidable challenge. In this particular scenario, the CFL number is incrementally increased from 1 to 1.0×10^4 within 800, 1000, 2000, and 2000 iterations for DDGP1, DDGP2, DDGP3, and DDGP4 schemes, respectively. Figure 22 presents the convergence histories of the aforementioned schemes, indicating an accelerated convergence as the CFL number rises. Notwithstanding a deceleration in the growth rate, the DDGP3 scheme encounters the CFL growth limit parameter multiple times during the computational process, with the threshold set at 0.1. For DDGP4, the calculation divergence after the residual drops to 3.5 orders of magnitude. Our hypothesis posits that minor unsteady flow features manifest or shock wave appears during the solution process, owing to the high resolution and low dissipation inherent in the high-order solution. It is important to note that further investigation is warranted to validate this assumption thoroughly. Figure 23 showcases the variation in the Krylov subspace during the computation of DDGP2 and DDGP3. Evidently, due to the gradual stabilization of the flow field, reducing the sub-iterative residual in GMRES becomes a formidable task. Even when the subspace reaches its maximum predetermined value, sub-iteration fails to meet the standards of convergence, leading to an increase in both single-step and total computation time. Owing to the intricate geometric characteristics, three distinct regions of flow separation occur within the flow field: one in the slat cove, another in the main wing cove, and a less conspicuous one above the flap. In addition, a velocity increase occurs on the leading edge of the slat due to a reduction in the flow area. These phenomena are evidently depicted in Fig. 24. Figure 25

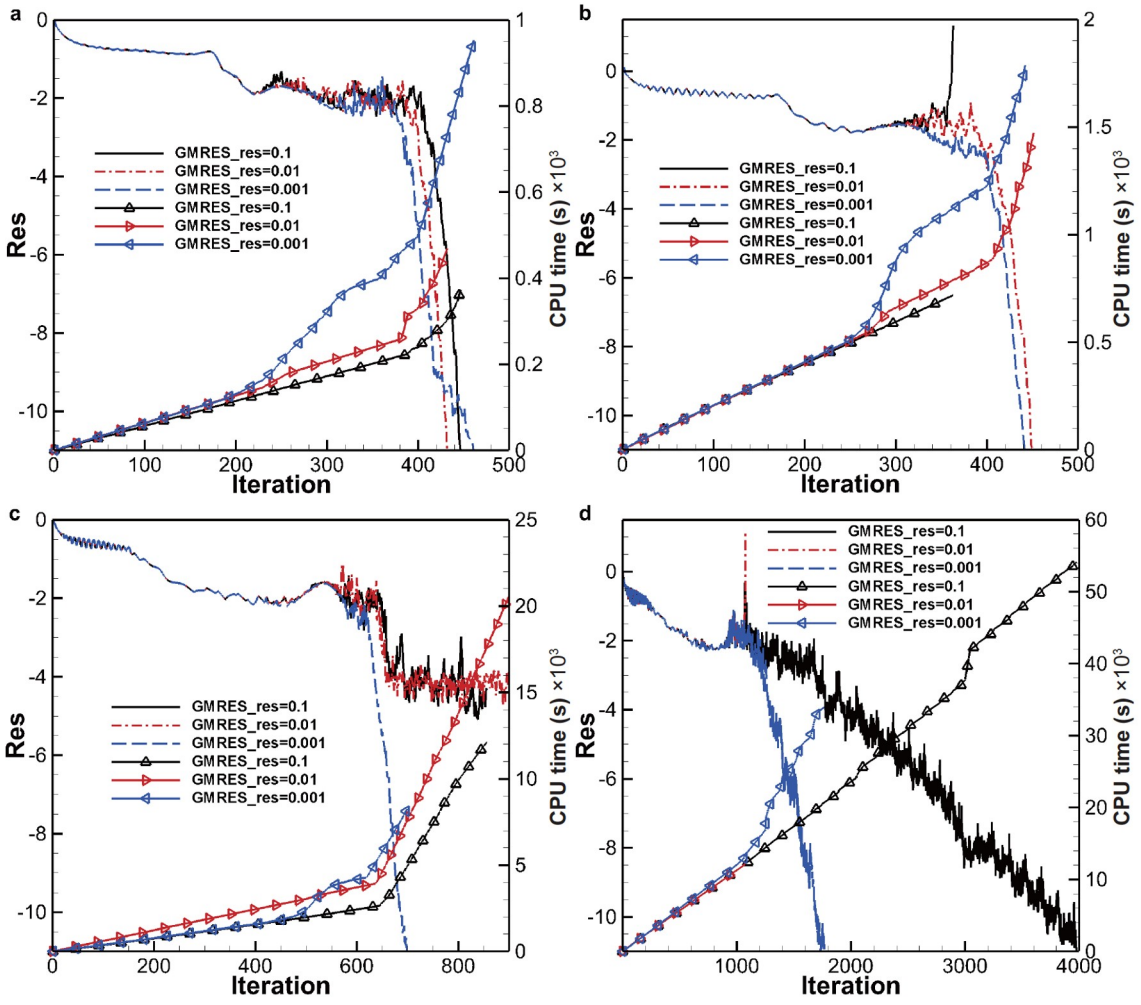

Figure 20 Influence of GMRES convergence tolerance to the convergence of calculations: **a** DDGP1; **b** DDGP2; **c** DDGP3; **d** DDGP4.

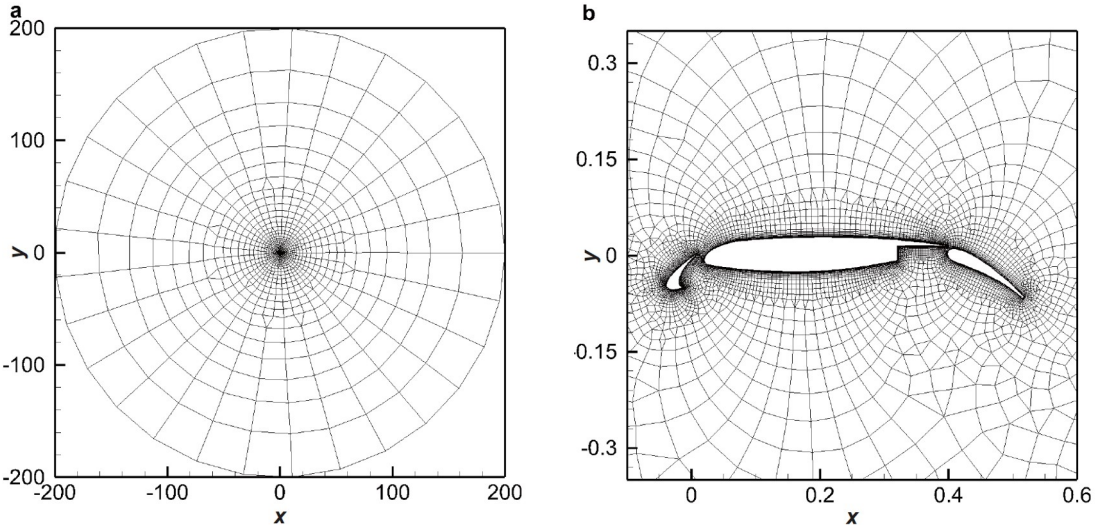

Figure 21 Hybrid mesh for turbulent flow over a 30P30N multi-element airfoil at attack angle of 5.5°: **a** computational domain; **b** zoom view.

gives the convergence history of the total lift coefficient with the order of DDG schemes. Similar to other test cases, as the calculation order increases, the computed value gradually approaches the reference value. A comparison between the distribution of the pressure coefficient on the airfoil surface, as calculated by the DDGP3 scheme, and the

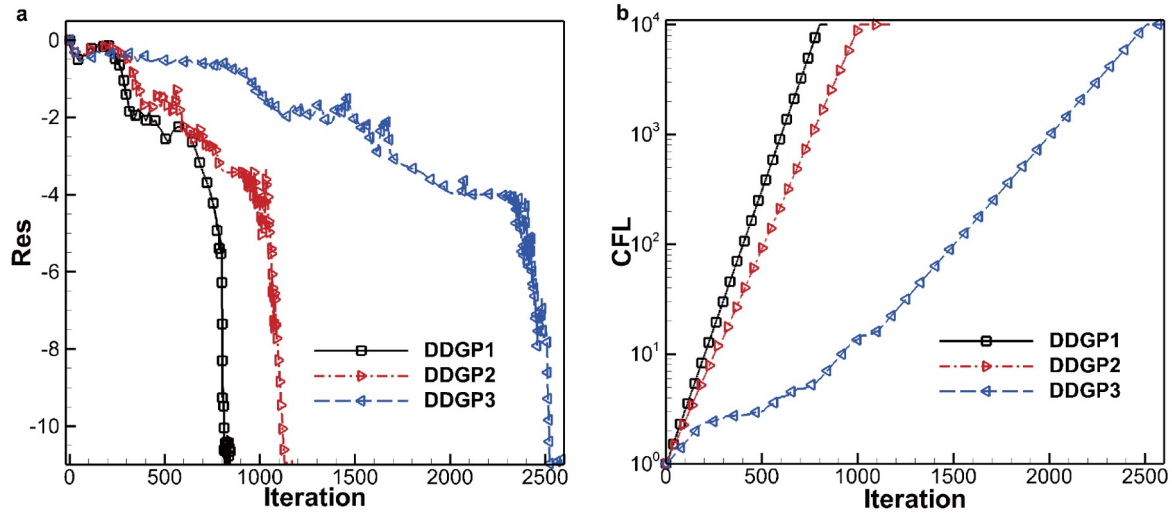

Figure 22 Comparison of convergence histories of DDGP1 to DDGP3 schemes: **a** residual-iteration; **b** CFL number- iteration.

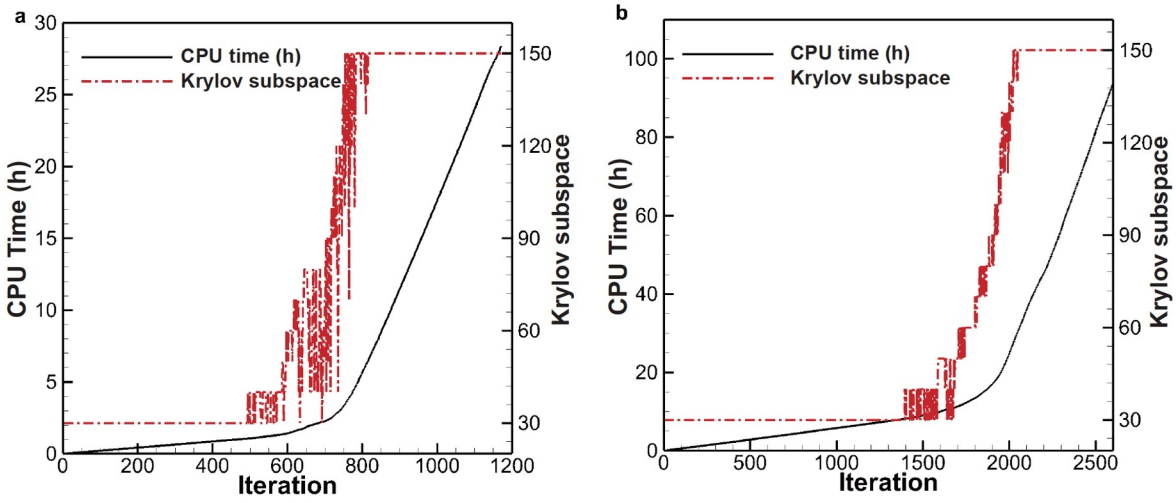

Figure 23 Krylov subspace in GMRES sub iteration: **a** DDGP2; **b** DDGP3.

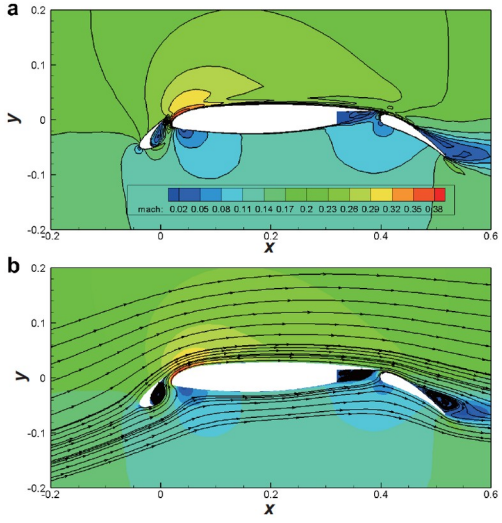

Figure 24 Mach number contour lines and streamlines in separation areas: **a** mach number contour lines; **b** streamlines.

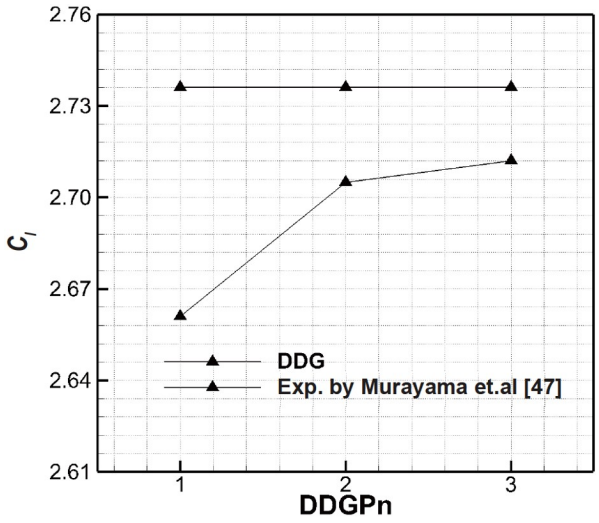

Figure 25 Convergence history of total lift coefficient with the order of DDG schemes.

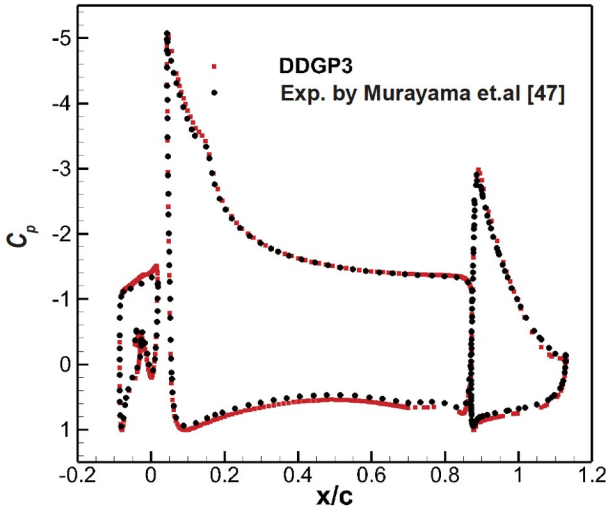

Figure 26 Pressure coefficient obtained by DDGP3 scheme.

corresponding experimental values [47] as shown in Fig. 26, expresses a high level of agreement.

6. Conclusions

This paper introduces a robust implicit high-order DG method for the numerical solution of compressible NS equations on arbitrary grids. The performance of this method is assessed using several benchmark test cases. Based on the obtained results, the following conclusions can be drawn.

(1) The proposed implicit method demonstrates nearly quadratic convergence in high-order calculations utilizing polynomial approximations ranging from P1 to P5. To strike a favorable balance between computational stability and efficiency, three strategies are employed. Firstly, simulations are mapped onto a reference domain, enhancing the reversibility of the mass matrix in implicit calculations and reducing the demand for high grid quality in high-order calculations. Secondly, an exact Jacobian matrix is employed for preconditioning and matrix-vector generation in the GMRES method, avoiding any approximations or simplifications that may introduce additional errors. This approach helps minimize error propagation. Lastly, an adaptive CFL number increasing strategy is implemented, dynamically adjusting the CFL number based on changes in the flow field. By adopting these strategies, the proposed method achieves both robustness and efficiency in high-order calculations.

(2) The efficiency of implicit calculations is influenced by various parameters. In low-order DG calculations, differences in these parameters do not significantly impact stability and efficiency. However, in higher-order DG methods, appropriate parameter settings are crucial to fully exploit the performance potential of the GMRES method. Numerical results indicate that the incremental Krylov subspace mode exhibits superior stability compared to the fixed Krylov subspace mode. Additionally, appropriate convergence criteria for GMRES can prevent excessive iterations during the iterative process. This study proposes a convergence criterion of 0.01 for the GMRES method, which is considered suitable for most cases. However, in complex flow problems, it is recommended to further decrease the convergence criterion to ensure computational stability.

(3) The combination of the DG method, GMRES solver, exact Jacobian matrix, adaptive CFL number, and reference domain demonstrates promising results in terms of robustness, convergence, and accuracy. The comprehensive analysis conducted in this study provides valuable insights and serves as a reference for implicit computations in high-order calculations. The findings demonstrate the effectiveness of the proposed method and underscore its potential for solving complex problems in CFD.

Appendix A

The inviscid Jacobian consists of two parts. One is the inviscid flux Jacobian of Roe scheme, the other is the inviscid domain Jacobian.

A.1 Inviscid flux Jacobian of Roe scheme

The inviscid flux $\mathbf{F}$ of Roe scheme is in the form of the conserved variables, $\mathbf{U}^{\mathrm{L}}$ and $\mathbf{U}^{\mathrm{R}}$. According to the chain rule, the derivative of the inviscid flux of Roe scheme can be divided into two parts, the derivative for the left state $\partial\mathbf{F}/\partial\mathbf{U}^{\mathrm{L}}$ and the derivative for the right state $\partial\mathbf{F}/\partial\mathbf{U}^{\mathrm{R}}$. The process of computing Roe flux Jacobian is as follows, and N_f, N_e is the interface number and element number in the entire computational domain, L, R is the left and right element of the interface. $N_t = N_q \times N_d$, where N_q is the number of the equations we solved and N_d is the degree of the high-order polynomial approximations.

do m = 1, N_f

do n = 1, N_t

(a) The derivative $\partial\mathbf{F}/\partial\mathbf{U}^{\mathrm{L}}$ is calculated first.

The conserved variables' derivatives, $\mathbf{U}_{\mathrm{d}}^{\mathrm{L}}$ and $\mathbf{U}_{\mathrm{d}}^{\mathrm{R}}$ for left state are first computed from

$$\mathbf{U}_{\mathrm{d}}^{\mathrm{L}} = \left.\frac{\partial\mathbf{U}_h^{\mathrm{L}}}{\partial\mathbf{U}^{\mathrm{L}}}\right|_k = \sum_{i=1}^{N}\mathbf{I}_{n,(k-1)\times N_d+i}b_i^{\mathrm{L}};\ \mathbf{U}_{\mathrm{d}}^{\mathrm{R}} = \left.\frac{\partial\mathbf{U}_h^{\mathrm{R}}}{\partial\mathbf{U}^{\mathrm{L}}}\right|_k = 0. \quad \text{(A1)}$$

The derivatives of u^{L}, v^{L}, p^{L}, $(\rho h)^{\mathrm{L}}$, v^{L}, q^{L}, a^{L}, and u^{R}, v^{R}, p^{R}, $(\rho h)^{\mathrm{R}}$, v^{R}, q^{R}, a^{R} are given by

$$
\begin{aligned}
&u_{\mathrm{d}}^{\mathrm{L}}=\left[(\rho u)_{\mathrm{d}}^{\mathrm{L}}-u^{\mathrm{L}}\rho_{\mathrm{d}}^{\mathrm{L}}\right]/\rho^{\mathrm{L}},\ v_{\mathrm{d}}^{\mathrm{L}}=\left[(\rho v)_{\mathrm{d}}^{\mathrm{L}}-v^{\mathrm{L}}\rho_{\mathrm{d}}^{\mathrm{L}}\right]/\rho^{\mathrm{L}},\\
&p_{\mathrm{d}}^{\mathrm{L}}=(\gamma-1)\left[(\rho e)_{\mathrm{d}}^{\mathrm{L}}-0.5\rho_{\mathrm{d}}^{\mathrm{L}}\left(u^{\mathrm{L}}u^{\mathrm{L}}+v^{\mathrm{L}}v^{\mathrm{L}}\right)-\rho^{\mathrm{L}}\left(u_{\mathrm{d}}^{\mathrm{L}}u^{\mathrm{L}}+v_{\mathrm{d}}^{\mathrm{L}}v^{\mathrm{L}}\right)\right]/\rho^{\mathrm{L}},\\
&(\rho h)_{\mathrm{d}}^{\mathrm{L}}=(\rho e)_{\mathrm{d}}^{\mathrm{L}}+p_{\mathrm{d}}^{\mathrm{L}},\ h_{\mathrm{d}}^{\mathrm{L}}=\left[(\rho h)_{\mathrm{d}}^{\mathrm{L}}-h^{\mathrm{L}}\rho_{\mathrm{d}}^{\mathrm{L}}\right]/\rho^{\mathrm{L}},\\
&v_{\mathrm{d}}^{\mathrm{L}}=\left[(\rho v)_{\mathrm{d}}^{\mathrm{L}}-v^{\mathrm{L}}\rho_{\mathrm{d}}^{\mathrm{L}}\right]/\rho^{\mathrm{L}},\ q_{\mathrm{d}}^{\mathrm{L}}=u_{\mathrm{d}}^{\mathrm{L}}n_x+v_{\mathrm{d}}^{\mathrm{L}}n_y,\ a_{\mathrm{d}}^{\mathrm{L}}=0.5\gamma\left(p_{\mathrm{d}}^{\mathrm{L}}-p^{\mathrm{L}}\rho_{\mathrm{d}}^{\mathrm{L}}/\rho^{\mathrm{L}}\right)/\rho^{\mathrm{L}}.
\end{aligned}
\tag{A2}
$$

$$
\begin{aligned}
&u_{\mathrm{d}}^{\mathrm{R}}=\left[(\rho u)_{\mathrm{d}}^{\mathrm{R}}-u^{\mathrm{R}}\rho_{\mathrm{d}}^{\mathrm{R}}\right]/\rho^{\mathrm{R}},\ v_{\mathrm{d}}^{\mathrm{R}}=\left[(\rho v)_{\mathrm{d}}^{\mathrm{R}}-v^{\mathrm{R}}\rho_{\mathrm{d}}^{\mathrm{R}}\right]/\rho^{\mathrm{R}},\\
&p_{\mathrm{d}}^{\mathrm{R}}=(\gamma-1)\left[(\rho e)_{\mathrm{d}}^{\mathrm{R}}-0.5\rho_{\mathrm{d}}^{\mathrm{R}}\left(u^{\mathrm{R}}u^{\mathrm{R}}+v^{\mathrm{R}}v^{\mathrm{R}}\right)-\rho^{\mathrm{R}}\left(u_{\mathrm{d}}^{\mathrm{R}}u^{\mathrm{R}}+v_{\mathrm{d}}^{\mathrm{R}}v^{\mathrm{R}}\right)\right]/\rho^{\mathrm{R}},\\
&(\rho h)_{\mathrm{d}}^{\mathrm{R}}=(\rho e)_{\mathrm{d}}^{\mathrm{R}}+p_{\mathrm{d}}^{\mathrm{R}},\ h_{\mathrm{d}}^{\mathrm{R}}=\left[(\rho h)_{\mathrm{d}}^{\mathrm{R}}-h^{\mathrm{R}}\rho_{\mathrm{d}}^{\mathrm{R}}\right]/\rho^{\mathrm{R}},\\
&v_{\mathrm{d}}^{\mathrm{R}}=\left[(\rho v)_{\mathrm{d}}^{\mathrm{R}}-v^{\mathrm{R}}\rho_{\mathrm{d}}^{\mathrm{R}}\right]/\rho^{\mathrm{R}},\ q_{\mathrm{d}}^{\mathrm{R}}=u_{\mathrm{d}}^{\mathrm{R}}n_x+v_{\mathrm{d}}^{\mathrm{R}}n_y,\ a_{\mathrm{d}}^{\mathrm{R}}=0.5\gamma\left(p_{\mathrm{d}}^{\mathrm{R}}-p^{\mathrm{R}}\rho_{\mathrm{d}}^{\mathrm{R}}/\rho^{\mathrm{R}}\right)/\rho^{\mathrm{R}}.
\end{aligned}
\tag{A3}
$$

The derivatives of Roe average values $\tilde{\rho}$, $\tilde{u}$, $\tilde{v}$, $\tilde{h}$, $\tilde{v},\tilde{a}$, and $\tilde{q}$ are also required

$$
\begin{aligned}
&\rho_{\mathrm{d}}^{\mathrm{o}}=0.5\left(\rho_{\mathrm{d}}^{\mathrm{R}}-\rho^{\mathrm{R}}\rho_{\mathrm{d}}^{\mathrm{L}}/\rho^{\mathrm{L}}\right)/\left(\rho^{\mathrm{o}}\rho^{\mathrm{L}}\right),\ \tilde{\rho}_{\mathrm{d}}=\rho_{\mathrm{d}}^{\mathrm{o}}\rho^{\mathrm{L}}+\rho_{t}\rho_{\mathrm{d}}^{\mathrm{R}},\\
&\tilde{u}_{\mathrm{d}}=\left(u_{\mathrm{d}}^{\mathrm{L}}+\rho_{\mathrm{d}}^{\mathrm{o}}u^{\mathrm{R}}+\rho^{\mathrm{o}}u_{\mathrm{d}}^{\mathrm{R}}-\tilde{u}\rho_{\mathrm{d}}^{\mathrm{o}}\right)/\left(1+\rho^{\mathrm{o}}\right),\\
&\tilde{v}_{\mathrm{d}}=\left(v_{\mathrm{d}}^{\mathrm{L}}+\rho_{\mathrm{d}}^{\mathrm{o}}v^{\mathrm{R}}+\rho^{\mathrm{o}}v_{\mathrm{d}}^{\mathrm{R}}-v\rho_{\mathrm{d}}^{\mathrm{o}}\right)/\left(1+\rho^{\mathrm{o}}\right),\\
&\tilde{v}_{\mathrm{d}}=\left(\tilde{v}_{\mathrm{d}}^{\mathrm{L}}+\rho_{\mathrm{d}}^{\mathrm{o}}\tilde{v}^{\mathrm{R}}+\rho^{\mathrm{o}}\tilde{v}_{\mathrm{d}}^{\mathrm{R}}-\tilde{v}\rho_{\mathrm{d}}^{\mathrm{o}}\right)/\left(1+\rho^{\mathrm{o}}\right),\\
&\tilde{h}_{\mathrm{d}}=\left(h_{\mathrm{d}}^{\mathrm{L}}+\rho_{\mathrm{d}}^{\mathrm{o}}h^{\mathrm{R}}+\rho^{\mathrm{o}}h_{\mathrm{d}}^{\mathrm{R}}-\tilde{h}\rho_{\mathrm{d}}^{\mathrm{o}}\right)/\left(1+\rho^{\mathrm{o}}\right),\\
&\tilde{a}_{\mathrm{d}}=0.5(\gamma-1)\left(\tilde{h}_{\mathrm{d}}-\tilde{u}\tilde{u}_{\mathrm{d}}-\tilde{v}\tilde{v}_{\mathrm{d}}\right)/\tilde{a},\ \tilde{q}_{\mathrm{d}}=\tilde{u}_{\mathrm{d}}n_x+\tilde{v}_{\mathrm{d}}n_y.
\end{aligned}
\tag{A4}
$$

The derivatives of Roe's dissipation, $(\Delta\rho)_{\mathrm{d}}$, $(\Delta u)_{\mathrm{d}}$, $(\Delta v)_{\mathrm{d}}$, $(\Delta v)_{\mathrm{d}}$, $(\Delta p)_{\mathrm{d}}$, $(\Delta q)_{\mathrm{d}}$ are computed from

$$
\begin{aligned}
&(\Delta\rho)_{\mathrm{d}}=\rho_{\mathrm{d}}^{\mathrm{R}}-\rho_{\mathrm{d}}^{\mathrm{L}},\ (\Delta u)_{\mathrm{d}}=u_{\mathrm{d}}^{\mathrm{R}}-u_{\mathrm{d}}^{\mathrm{L}},\\
&(\Delta v)_{\mathrm{d}}=v_{\mathrm{d}}^{\mathrm{R}}-v_{\mathrm{d}}^{\mathrm{L}},\ (\Delta v)_{\mathrm{d}}=v_{\mathrm{d}}^{\mathrm{R}}-v_{\mathrm{d}}^{\mathrm{L}},\\
&(\Delta p)_{\mathrm{d}}=p_{\mathrm{d}}^{\mathrm{R}}-p_{\mathrm{d}}^{\mathrm{L}},\ (\Delta q)_{\mathrm{d}}=q_{\mathrm{d}}^{\mathrm{R}}-q_{\mathrm{d}}^{\mathrm{L}}.
\end{aligned}
\tag{A5}
$$

The derivatives of the eigenvalues of Jacobian $\lambda_{1,\mathrm{d}}$, $\lambda_{2,\mathrm{d}}$, $\lambda_{3,\mathrm{d}}$, and the derivatives of the entropy fix value $\varepsilon_{1,\mathrm{d}}$, $\varepsilon_{2,\mathrm{d}}$, $\varepsilon_{3,\mathrm{d}}$ are written as

$$
\begin{aligned}
&\lambda_{1,\mathrm{d}}=\begin{cases}\tilde{q}_{\mathrm{d}}, & \text{if } \tilde{q}\geqslant 0,\\ -\tilde{q}_{\mathrm{d}}, & \text{if } \tilde{q}<0,\end{cases}\\
&\lambda_{2,\mathrm{d}}=\begin{cases}\tilde{q}_{\mathrm{d}}+\tilde{c}_{\mathrm{d}}, & \text{if } \tilde{q}+\tilde{c}\geqslant 0,\\ -(\tilde{q}_{\mathrm{d}}+\tilde{c}_{\mathrm{d}}), & \text{if } \tilde{q}+\tilde{c}<0,\end{cases}\\
&\lambda_{3,\mathrm{d}}=\begin{cases}\tilde{q}_{\mathrm{d}}-\tilde{c}_{\mathrm{d}}, & \text{if } \tilde{q}-\tilde{c}\geqslant 0,\\ -(\tilde{q}_{\mathrm{d}}-\tilde{c}_{\mathrm{d}}), & \text{if } \tilde{q}-\tilde{c}<0,\end{cases}
\end{aligned}
\tag{A6}
$$

$$
\begin{aligned}
&\varepsilon_{1,\mathrm{d}}=\begin{cases}\tilde{q}_{\mathrm{d}}-q_{\mathrm{d}}^{\mathrm{L}}, & \text{if } \varepsilon_1=\tilde{q}-q^{\mathrm{L}},\\ q_{\mathrm{d}}^{\mathrm{R}}-\tilde{q}_{\mathrm{d}}, & \text{if } \varepsilon_1=q^{\mathrm{R}}-\tilde{q},\\ 0, & \text{if } \varepsilon_1=0,\end{cases}\\
&\varepsilon_{2,\mathrm{d}}=\begin{cases}(\tilde{q}_{\mathrm{d}}+\tilde{c}_{\mathrm{d}})-\left(q_{\mathrm{d}}^{\mathrm{L}}+c_{\mathrm{d}}^{\mathrm{L}}\right), & \text{if } \varepsilon_2=(\tilde{q}+\tilde{c})-(q^{\mathrm{L}}+c^{\mathrm{L}}),\\ \left(q_{\mathrm{d}}^{\mathrm{R}}+c_{\mathrm{d}}^{\mathrm{R}}\right)-(\tilde{q}_{\mathrm{d}}+\tilde{c}_{\mathrm{d}}), & \text{if } \varepsilon_2=(q^{\mathrm{R}}+c^{\mathrm{R}})-(\tilde{q}+\tilde{c}),\\ 0, & \text{if } \varepsilon_2=0,\end{cases}\\
&\varepsilon_{3,\mathrm{d}}=\begin{cases}(\tilde{q}_{\mathrm{d}}-\tilde{c}_{\mathrm{d}})-\left(q_{\mathrm{d}}^{\mathrm{L}}-c_{\mathrm{d}}^{\mathrm{L}}\right), & \text{if } \varepsilon_3=(\tilde{q}-\tilde{c})-(q^{\mathrm{L}}-c^{\mathrm{L}}),\\ \left(q_{\mathrm{d}}^{\mathrm{R}}-c_{\mathrm{d}}^{\mathrm{R}}\right)-(\tilde{q}_{\mathrm{d}}-\tilde{c}_{\mathrm{d}}), & \text{if } \varepsilon_3=(q^{\mathrm{R}}-c^{\mathrm{R}})-(\tilde{q}-\tilde{c}),\\ 0, & \text{if } \varepsilon_3=0.\end{cases}
\end{aligned}
\tag{A7}
$$

According to the relationship of the eigenvalues of Jacobian and the entropy fix value, we get the final derivatives of the eigenvalues of Jacobian $\lambda_{1,\mathrm{d}}$, $\lambda_{2,\mathrm{d}}$, $\lambda_{3,\mathrm{d}}$

$$
\lambda_{i,\mathrm{d}}=\begin{cases}\varepsilon_{i,\mathrm{d}}, & \text{if } \lambda_i\leq\varepsilon_i,\\ \lambda_{i,\mathrm{d}}, & \text{if } \lambda_i>\varepsilon.\end{cases}
\tag{A8}
$$

The derivatives of the parameters of Roe flux $a_{1,\mathrm{d}}$, $a_{2,\mathrm{d}}$, $a_{3,\mathrm{d}}$, $a_{4,\mathrm{d}}$, $a_{5,\mathrm{d}}$, $a_{6,\mathrm{d}}$, $a_{7,\mathrm{d}}$ can be given by

$$
\begin{aligned}
a_{1,\mathrm{d}} &= \lambda_{1,\mathrm{d}}\left(\Delta\rho-\frac{\Delta p}{\tilde{c}^2}\right)+\lambda_1\left[(\Delta\rho)_{\mathrm{d}}-\left(\frac{(\Delta p)_{\mathrm{d}}}{\tilde{c}^2}-\frac{2\Delta p\cdot\tilde{c}_{\mathrm{d}}}{\tilde{c}^3}\right)\right],\\
a_{2,\mathrm{d}} &= \lambda_{2,\mathrm{d}}\frac{(\Delta p+\tilde{\rho}\,\tilde{c}\Delta q)}{2\tilde{c}^2}+\lambda_2\frac{\left[(\Delta p)_{\mathrm{d}}+\tilde{\rho}_{\mathrm{d}}\,\tilde{c}\Delta q+\tilde{\rho}\,\tilde{c}_{\mathrm{d}}\Delta q+\tilde{\rho}\,\tilde{c}(\Delta q)_{\mathrm{d}}\right]}{2\tilde{c}^2}-\lambda_2\frac{(\Delta p+\tilde{\rho}\,\tilde{c}\Delta q)\cdot\tilde{c}_{\mathrm{d}}}{\tilde{c}^3},\\
a_{3,\mathrm{d}} &= \lambda_{3,\mathrm{d}}\frac{(\Delta p-\tilde{\rho}\,\tilde{c}\Delta q)}{2\tilde{c}^2}+\lambda_2\frac{\left[(\Delta p)_{\mathrm{d}}-\tilde{\rho}_{\mathrm{d}}\,\tilde{c}\Delta q-\tilde{\rho}\,\tilde{c}_{\mathrm{d}}\Delta q-\tilde{\rho}\,\tilde{c}(\Delta q)_{\mathrm{d}}\right]}{2\tilde{c}^2}-\lambda_2\frac{(\Delta p-\tilde{\rho}\,\tilde{c}\Delta q)\cdot\tilde{c}_{\mathrm{d}}}{\tilde{c}^3},\\
a_{4,\mathrm{d}} &= a_{1,\mathrm{d}}+a_{2,\mathrm{d}}+a_{3,\mathrm{d}},\\
a_{5,\mathrm{d}} &= \tilde{c}_{\mathrm{d}}(a_2+a_3)+\tilde{c}\left(a_{2,\mathrm{d}}+a_{3,\mathrm{d}}\right),
\end{aligned}
\tag{A9}
$$

$$
\begin{aligned}
a_{6,\mathrm{d}} &= \lambda_{1,\mathrm{d}}(\tilde{\rho}\,\Delta u-\tilde{\rho}\,\Delta q\cdot n_x)+\lambda_1\left\{\tilde{\rho}_{\mathrm{d}}\Delta u+\tilde{\rho}\,(\Delta u)_{\mathrm{d}}-\left[\tilde{\rho}_{\mathrm{d}}\Delta q+\tilde{\rho}\,(\Delta q)_{\mathrm{d}}\right]\cdot n_x\right\},\\
a_{7,\mathrm{d}} &= \lambda_{1,\mathrm{d}}\left(\tilde{\rho}\,\Delta v-\tilde{\rho}\,\Delta q\cdot n_y\right)+\lambda_1\left\{\tilde{\rho}_{\mathrm{d}}\Delta v+\tilde{\rho}\,(\Delta v)_{\mathrm{d}}-\left[\tilde{\rho}_{\mathrm{d}}\Delta q+\tilde{\rho}\,(\Delta q)_{\mathrm{d}}\right]\cdot n_y\right\},\\
a_{8,\mathrm{d}} &= \lambda_{1,\mathrm{d}}\left(\tilde{\rho}\,\Delta v+\tilde{v}\Delta\rho-\frac{\tilde{v}\Delta p}{\tilde{c}^2}\right)+\lambda_1\left[\tilde{\rho}_{\mathrm{d}}\Delta v+\tilde{v}_{\mathrm{d}}\Delta\rho+\tilde{\rho}\,(\Delta v)_{\mathrm{d}}+\tilde{v}(\Delta\rho)_{\mathrm{d}}\right]-\lambda_1\left[\frac{\tilde{v}_{\mathrm{d}}\Delta p+\tilde{v}(\Delta p)_{\mathrm{d}}}{\tilde{c}^2}-\frac{2\tilde{v}\Delta p\cdot\tilde{c}_{\mathrm{d}}}{\tilde{c}^3}\right].
\end{aligned}
\tag{A10}
$$

According to the above derivatives, we get $b_{1,\mathrm{d}}$, $b_{2,\mathrm{d}}$, $b_{3,\mathrm{d}}$, $b_{4,\mathrm{d}}$, $b_{5,\mathrm{d}}$ for calculating $\partial\mathbf{F}/\partial\mathbf{U}^{\mathrm{L}}$

$$
\begin{aligned}
b_{1,\mathrm{d}} &= a_{4,\mathrm{d}},\\
b_{2,\mathrm{d}} &= \tilde{u}_{\mathrm{d}}a_4+\tilde{u}a_{4,\mathrm{d}}+a_{5,\mathrm{d}}n_x,\\
b_{3,\mathrm{d}} &= \tilde{v}_{\mathrm{d}}a_4+\tilde{v}a_{4,\mathrm{d}}+a_{5,\mathrm{d}}n_y,\\
b_{4,\mathrm{d}} &= \tilde{h}_{\mathrm{d}}a_4+\tilde{h}a_{4,\mathrm{d}}+\tilde{q}_{\mathrm{d}}a_5+\tilde{q}a_{5,\mathrm{d}}+\tilde{u}_{\mathrm{d}}a_6+\tilde{u}a_{6,\mathrm{d}}+\tilde{v}_{\mathrm{d}}a_7+\tilde{v}a_{7,\mathrm{d}}-\left(2a_1\tilde{c}\tilde{c}_{\mathrm{d}}+a_{1,\mathrm{d}}\tilde{c}^2\right)/(\gamma-1),\\
b_{5,\mathrm{d}} &= \tilde{v}_{\mathrm{d}}(a_2+a_3)+\tilde{v}\left(a_{2,\mathrm{d}}+a_{3,\mathrm{d}}\right)+a_{8,\mathrm{d}}.
\end{aligned}
\tag{A11}
$$

Finally, the derivative $\partial\mathbf{F}/\partial\mathbf{U}^{\mathrm{L}}$ is given as

$$
\frac{\partial\mathbf{F}}{\partial\mathbf{U}^{\mathrm{L}}}=\begin{bmatrix}
\frac{1}{2}\left(\rho_{\mathrm{d}}^{\mathrm{L}}q^{\mathrm{L}}+\rho^{\mathrm{L}}q_{\mathrm{d}}^{\mathrm{L}}+\rho_{\mathrm{d}}^{\mathrm{R}}q^{\mathrm{R}}+\rho^{\mathrm{R}}q_{\mathrm{d}}^{\mathrm{R}}-b_{1,\mathrm{d}}\right)\\
\frac{1}{2}\left[(\rho u)_{\mathrm{d}}^{\mathrm{L}}q^{\mathrm{L}}+(\rho u)^{\mathrm{L}}q_{\mathrm{d}}^{\mathrm{L}}+(\rho u)_{\mathrm{d}}^{\mathrm{R}}q^{\mathrm{R}}+(\rho u)^{\mathrm{R}}q_{\mathrm{d}}^{\mathrm{R}}+\left(p_{\mathrm{d}}^{\mathrm{L}}+p_{\mathrm{d}}^{\mathrm{R}}\right)n_x-b_{2,\mathrm{d}}\right]\\
\frac{1}{2}\left[(\rho v)_{\mathrm{d}}^{\mathrm{L}}q^{\mathrm{L}}+(\rho v)^{\mathrm{L}}q_{\mathrm{d}}^{\mathrm{L}}+(\rho v)_{\mathrm{d}}^{\mathrm{R}}q^{\mathrm{R}}+(\rho v)^{\mathrm{R}}q_{\mathrm{d}}^{\mathrm{R}}+\left(p_{\mathrm{d}}^{\mathrm{L}}+p_{\mathrm{d}}^{\mathrm{R}}\right)n_y-b_{3,\mathrm{d}}\right]\\
\frac{1}{2}\left[(\rho h)_{\mathrm{d}}^{\mathrm{L}}q^{\mathrm{L}}+(\rho h)^{\mathrm{L}}q_{\mathrm{d}}^{\mathrm{L}}+(\rho h)_{\mathrm{d}}^{\mathrm{R}}q^{\mathrm{R}}+(\rho h)^{\mathrm{R}}q_{\mathrm{d}}^{\mathrm{R}}-b_{4,\mathrm{d}}\right]\\
\frac{1}{2}\left[(\rho v)_{\mathrm{d}}^{\mathrm{L}}q^{\mathrm{L}}+(\rho v)^{\mathrm{L}}q_{\mathrm{d}}^{\mathrm{L}}+(\rho v)_{\mathrm{d}}^{\mathrm{R}}q^{\mathrm{R}}+(\rho v)^{\mathrm{R}}q_{\mathrm{d}}^{\mathrm{R}}-b_{5,\mathrm{d}}\right]
\end{bmatrix}.
\tag{A12}
$$

(b) After that, the derivative $\partial\mathbf{F}/\partial\mathbf{U}^{\mathrm{R}}$can be obtained easily.

The difference between the Roe flux derivatives for the right state and the Roe flux derivatives for the left state is the conserved variables derivatives. In this part, the conserved variables derivatives of $\mathbf{U}_h^{\mathrm{R}}$ and $\mathbf{U}_h^{\mathrm{L}}$ for right state are written as

$$
\begin{aligned}
\mathbf{U}_{\mathrm{d}}^{\mathrm{R}} &= \left.\frac{\partial\mathbf{U}_h^{\mathrm{R}}}{\partial\mathbf{U}^{\mathrm{R}}}\right|_k=\sum_{i=1}^{N}\mathbf{I}_{n,(k-1)\times N_d+i}b_i^{\mathrm{R}},\\
\mathbf{U}_{\mathrm{d}}^{\mathrm{L}} &= \left.\frac{\partial\mathbf{U}_h^{\mathrm{L}}}{\partial\mathbf{U}^{\mathrm{R}}}\right|_k=0.
\end{aligned}
\tag{A13}
$$

Then, we can use the same process from Eq. (A2) to Eq. (A11) in step 1 to get the derivative $\partial\mathbf{F}/\partial\mathbf{U}^{\mathrm{R}}$.

$$\frac{\partial \mathbf{F}}{\partial \mathbf{U}^{\mathrm{R}}}=\begin{bmatrix} \frac{1}{2}\left(\rho_{\mathrm{d}}^{\mathrm{L}}q^{\mathrm{L}}+\rho^{\mathrm{L}}q_{\mathrm{d}}^{\mathrm{L}}+\rho_{\mathrm{d}}^{\mathrm{R}}q^{\mathrm{R}}+\rho^{\mathrm{R}}q_{\mathrm{d}}^{\mathrm{R}}-b_{1,\mathrm{d}}\right) \\ \frac{1}{2}\left[(\rho u)_{\mathrm{d}}^{\mathrm{L}}q^{\mathrm{L}}+(\rho u)^{\mathrm{L}}q_{\mathrm{d}}^{\mathrm{L}}+(\rho u)_{\mathrm{d}}^{\mathrm{R}}q^{\mathrm{R}}+(\rho u)^{\mathrm{R}}q_{\mathrm{d}}^{\mathrm{R}}+\left(p_{\mathrm{d}}^{\mathrm{L}}+p_{\mathrm{d}}^{\mathrm{R}}\right)n_x-b_{2,\mathrm{d}}\right] \\ \frac{1}{2}\left[(\rho v)_{\mathrm{d}}^{\mathrm{L}}q^{\mathrm{L}}+(\rho v)^{\mathrm{L}}q_{\mathrm{d}}^{\mathrm{L}}+(\rho v)_{d}^{R}q^{\mathrm{R}}+(\rho v)^{\mathrm{R}}q_{\mathrm{d}}^{\mathrm{R}}+\left(p_{\mathrm{d}}^{\mathrm{L}}+p_{\mathrm{d}}^{\mathrm{R}}\right)n_y-b_{3,\mathrm{d}}\right] \\ \frac{1}{2}\left[(\rho h)_{\mathrm{d}}^{\mathrm{L}}q^{\mathrm{L}}+(\rho h)^{\mathrm{L}}q_{\mathrm{d}}^{\mathrm{L}}+(\rho h)_{\mathrm{d}}^{\mathrm{R}}q^{\mathrm{R}}+(\rho h)^{\mathrm{R}}q_{\mathrm{d}}^{\mathrm{R}}-b_{4,\mathrm{d}}\right] \\ \frac{1}{2}\left[(\rho v)_{\mathrm{d}}^{\mathrm{L}}q^{\mathrm{L}}+(\rho v)^{\mathrm{L}}q_{\mathrm{d}}^{\mathrm{L}}+(\rho v)_{\mathrm{d}}^{\mathrm{R}}q^{\mathrm{R}}+(\rho v)^{\mathrm{R}}q_{\mathrm{d}}^{\mathrm{R}}-b_{5,\mathrm{d}}\right] \end{bmatrix}. \tag{A14}$$

(c) Combine $\partial\mathbf{F}/\partial\mathbf{U}^{\mathrm{L}}$ and $\partial\mathbf{F}/\partial\mathbf{U}^{\mathrm{R}}$ to get the inviscid flux Jacobians.

In the last step, using Eqs. (A12) and (A14) to combine the inviscid flux Jacobians, **D**, **L**, and **U**.

do j = 1, N_q
do k = 1, N_d
$kj = (j-1)\times N_d + k$

$$\begin{aligned} \mathbf{D}_{\mathrm{inv}}(kj,n,l) &= \mathbf{D}_{\mathrm{inv}}(kj,n,l)+\partial\mathbf{F}/\partial\mathbf{U}^{\mathrm{L}}b_k^{\mathrm{L}}\Gamma_e, \\ \mathbf{D}_{\mathrm{inv}}(kj,n,r) &= \mathbf{D}_{\mathrm{inv}}(kj,n,r)-\partial\mathbf{F}/\partial\mathbf{U}^{\mathrm{R}}b_k^{\mathrm{R}}\Gamma_e, \\ \mathbf{L}_{\mathrm{inv}}(kj,n,m) &= \mathbf{L}_{\mathrm{inv}}(kj,n,m)-\partial\mathbf{F}/\partial\mathbf{U}^{\mathrm{L}}b_k^{\mathrm{R}}\Gamma_e, \\ \mathbf{U}_{\mathrm{inv}}(kj,n,m) &= \mathbf{U}_{\mathrm{inv}}(kj,n,m)+\partial\mathbf{F}/\partial\mathbf{U}^{\mathrm{R}}b_k^{\mathrm{L}}\Gamma_e. \end{aligned} \tag{A15}$$

end do
end do
end do
end do

A.2 Inviscid domain Jacobian

Compared with the inviscid flux Jacobian of Roe scheme, the inviscid domain Jacobian can be got more easily because the inviscid domain integral is only contributed to the diagonal matrix **D**. The process of these two parts is almost the same, and the details are as follows:

do m = 1, N_e
do n = 1, N_t

(d) Get the conserved variables derivative $\mathbf{U}_{\mathrm{d}}$.

$$\mathbf{U}_{\mathrm{d}}=\left.\frac{\partial\mathbf{U}_h}{\partial\mathbf{U}}\right|_k=\sum_{i=1}^{N}\mathbf{I}_{n,(k-1)\times N_d+i}b_i. \tag{A16}$$

(e) Use chain rule to get the domain derivatives.

The form of related derivatives $u_{\mathrm{d}}, v_{\mathrm{d}}, v_{\mathrm{d}}, p_{\mathrm{d}}, (\rho h)_{\mathrm{d}}$ are given as

$$\begin{aligned} &u_{\mathrm{d}}=\left[(\rho u)_{\mathrm{d}}-u\rho_{\mathrm{d}}\right]/\rho,\quad v_{\mathrm{d}}=\left[(\rho v)_{\mathrm{d}}-v\rho_{\mathrm{d}}\right]/\rho, \\ &p_{\mathrm{d}}=(\gamma-1)\left[(\rho e)_{\mathrm{d}}-0.5\rho_{\mathrm{d}}\left(u^2+v^2\right)-\rho(uu_{\mathrm{d}}+vv_{\mathrm{d}})\right]/\rho, \\ &(\rho h)_{\mathrm{d}}=(\rho e)_{\mathrm{d}}+p_{\mathrm{d}}. \end{aligned} \tag{A17}$$

Then, the derivatives of flux at the x and y directions are computed from

$$\mathbf{F}_{1\mathrm{d}}=\begin{bmatrix} \rho_{\mathrm{d}}u+\rho u_{\mathrm{d}} \\ (\rho u)_{\mathrm{d}}u+(\rho u)u_{\mathrm{d}}+p_{\mathrm{d}} \\ (\rho v)_{\mathrm{d}}u+(\rho v)u_{\mathrm{d}} \\ (\rho h)_{\mathrm{d}}u+(\rho h)u_{\mathrm{d}} \\ (\rho v)_{\mathrm{d}}u+(\rho v)u_{\mathrm{d}} \end{bmatrix},\quad \mathbf{F}_{2\mathrm{d}}=\begin{bmatrix} \rho_{\mathrm{d}}v+\rho v_{\mathrm{d}} \\ (\rho v)_{\mathrm{d}}u+(\rho v)u_{\mathrm{d}} \\ (\rho v)_{\mathrm{d}}v+(\rho v)v_{\mathrm{d}}+p_{\mathrm{d}} \\ (\rho h)_{\mathrm{d}}v+(\rho h)v_{\mathrm{d}} \\ (\rho v)_{\mathrm{d}}v+(\rho v)v_{\mathrm{d}} \end{bmatrix}. \tag{A18}$$

(f) Combine the inviscid domain Jacobians to the matrix **D**.

The same as the process of inviscid flux Jacobian, using Eq. (A18) to combine the inviscid domain Jacobian

do j = 1, N_q
do k = 1, N_d
$kj = (j-1)\times N_d + k$

$$\mathbf{D}_{\mathrm{inv}}(kj,n,m)=\mathbf{D}_{\mathrm{inv}}(kj,n,m)-\left(\mathbf{F}_{1\mathrm{d}}\frac{\partial b_k}{\partial\xi}\frac{\partial\xi}{\partial x}+\mathbf{F}_{2\mathrm{d}}\frac{\partial b_k}{\partial\eta}\frac{\partial\eta}{\partial y}\right)\Omega. \tag{A19}$$

end do
end do
end do
end do

Conflict of interest *On behalf of all authors, the corresponding author states that there is no conflict of interest.*

Author contributions *Jia Yan carried out the work and wrote the first draft of the manuscript. Xiaoquan Yang helped organize the manuscript. Peifen Weng revised and edited the final version.*

Acknowledgements *This work was supported by the National Natural Science Foundation of China (Grant No. 12102247) and the Technology Development Program (Grant No. JCKY2022110C119).*

1 H. Luo, J. D. Baum, and R. Löhner, A discontinuous Galerkin method based on a Taylor basis for the compressible flows on arbitrary grids, J. Comput. Phys. **227**, 8875 (2008).
2 Z. H. Jiang, C. Yan, and J. Yu, Implicit high-order discontinuous Galerkin method with HWENO type limiters for steady viscous flow

simulations, Acta Mech. Sin. **29**, 526 (2013).
3 Z. H. Jiang, C. Yan, and J. Yu, A simple a posteriori indicator for discontinuous Galerkin method on unstructured grids, Acta Mech. Sin. **39**, 322296 (2023).
4 P. Delorme, P. Mazet, C. Peyret, and Y. Ventribout, Computational aeroacoustics applications based on a discontinuous Galerkin method, Comptes Rendus Mécanique **333**, 676 (2005).
5 J. Zhao, and H. Tang, Runge-Kutta discontinuous Galerkin methods for the special relativistic magnetohydrodynamics, J. Comput. Phys. **343**, 33 (2017).
6 S. Hennemann, A. M. Rueda-Ramírez, F. J. Hindenlang, and G. J. Gassner, A provably entropy stable subcell shock capturing approach for high order split form DG for the compressible Euler equations, J. Comput. Phys. **426**, 109935 (2021).
7 X. He, K. Wang, T. Liu, Y. Feng, B. Zhang, W. Yuan, and X. Wang, HODG: High-order discontinuous Galerkin methods for solving compressible Euler and Navier-Stokes equations—An open-source component-based development framework, Comput. Phys. Commun. **286**, 108660 (2023).
8 Y. Jiang, and H. Liu, Invariant-region-preserving DG methods for multi-dimensional hyperbolic conservation law systems, with an application to compressible Euler equations, J. Comput. Phys. **373**, 385 (2018).
9 D. N. Arnold, An interior penalty finite element method with discontinuous elements, SIAM J. Numer. Anal. **19**, 742 (1982).
10 B. Cockburn, and C. W. Shu, The local discontinuous Galerkin method for time-dependent convection-diffusion systems, SIAM J. Numer. Anal. **35**, 2440 (1998).
11 J. Peraire, and P. O. Persson, The compact discontinuous Galerkin (CDG) method for elliptic problems, SIAM J. Sci. Comput. **30**, 1806 (2008).
12 B. V. Leer, M. Lo, and M. V. Raalte, in A discontinuous Galerkin method for diffusion based on recovery: Proceedings of the 18th AIAA Computational Fluid Dynamics Conference, Miami, 2007.
13 H. Luo, L. Luo, R. Nourgaliev, V. A. Mousseau, and N. Dinh, A reconstructed discontinuous Galerkin method for the compressible Navier-Stokes equations on arbitrary grids, J. Comput. Phys. **229**, 6961 (2010).
14 F. Bassi, A. Crivellini, S. Rebay, and M. Savini, Discontinuous Galerkin solution of the Reynolds-averaged Navier-Stokes and k-ω turbulence model equations, Comput. Fluids **34**, 507 (2005).
15 H. Liu, and J. Yan, The direct discontinuous Galerkin (DDG) methods for diffusion problems, SIAM J. Numer. Anal. **47**, 675 (2009).
16 H. Liu, and J. Yan, The direct discontinuous Galerkin (DDG) method for diffusion with interface corrections, Commun. Comput. Phys. **8**, 541 (2010).
17 J. Cheng, X. Yang, X. Liu, T. Liu, and H. Luo, A direct discontinuous Galerkin method for the compressible Navier-Stokes equations on arbitrary grids, J. Comput. Phys. **327**, 484 (2016).
18 J. Cheng, X. Liu, X. Yang, T. Liu, and H. Luo, in A direct discontinuous Galerkin method for computation of turbulent flows on hybrid grids: Proceedings of the 46th AIAA Fluid Dynamics Conference, Washington, 2016.
19 J. Jaśkowiec, Discontinuous Galerkin method on reference domain, Comput. Assist. Methods Eng. Sci. **22**, 177 (2017).
20 J. Jaśkowiec, Very high order discontinuous Galerkin method in elliptic problems, Comput. Mech. **62**, 1 (2018).
21 H. Luo, J. D. Baum, and R. Löhner, A fast, matrix-free implicit method for compressible flows on unstructured grids, J. Comput. Phys. **146**, 664 (1998).
22 H. Luo, H. Segawa, and M. R. Visbal, An implicit discontinuous Galerkin method for the unsteady compressible Navier-Stokes equations, Comput. Fluids **53**, 133 (2012).
23 H. Ying, and L. Hao, Preconditioned GMRES method for a class of Toeplitz linear systems, Math. Numer. Sin. **43**, 177 (2021).
24 S. Correnty, E. Jarlebring, and K. M. Soodhalter, Preconditioned infinite GMRES for parameterized linear systems, SIAM J. Sci. Comput. S120 (2023).
25 Y. Saad, Iterative Methods for Sparse Linear Systems, 2nd ed. (Soc. for Industrial and Applied Mathematics, Philadelphia, 2003).
26 F. Bassi, and S. Rebay, GMRES discontinuous Galerkin solution of the compressible Navier-Stokes equations, in: Discontinuous Galerkin Methods (Springer, Berlin, Heidelberg, 2000), pp. 197-208.
27 P. O. Persson, and J. Peraire, Newton-GMRES preconditioning for discontinuous Galerkin discretizations of the Navier-Stokes equations, SIAM J. Sci. Comput. **30**, 2709 (2008).
28 M. J. Zahr, and P. O. Persson, in Performance tuning of Newton-GMRES methods for discontinuous Galerkin discretizations of the Navier-Stokes equations: Proceedings of the 21st AIAA Computational Fluid Dynamics Conference, San Diego, 2013.
29 T. L. Tysinger, and D. A. Caughey, Alternating direction implicit methods for the Navier-Stokes equations, AIAA J. **30**, 2158 (1992).
30 J. Liu, J. Chen, Z. Zhang, Y. Yang, and Z. Xiao, Assessment of a new hybrid-SSOR implicit temporal scheme for turbulent flows across a wide range of Mach numbers, Acta Mech. Sin. **39**, 322398 (2023).
31 N. Nigro, M. Storti, S. Idelsohn, and T. Tezduyar, Physics based GMRES preconditioner for compressible and incompressible Navier-Stokes equations, Comput. Methods Appl. Mech. Eng. **154**, 203 (1998).
32 X. Yang, C. Jian, C. Wang, and H. Luo, in A fast, implicit discontinuous Galerkin method based on analytical Jacobians for the compressible Navier-Stokes equations: Proceedings of the 54th AIAA Aerospace Sciences Meeting, San Diego, 2016.
33 X. Yang, J. Cheng, H. Luo, and Q. Zhao, Robust implicit direct discontinuous Galerkin method for simulating the compressible turbulent flows, AIAA J. **57**, 1113 (2019).
34 S. C. Eisenstat, and H. F. Walker, Choosing the forcing terms in an inexact Newton method, SIAM J. Sci. Comput. **17**, 16 (1996).
35 H. B. An, Z. Y. Mo, and X. P. Liu, A choice of forcing terms in inexact Newton method, J. Comput. Appl. Math. **200**, 47 (2007).
36 K. Lund, Adaptively restarted block Krylov subspace methods with low-synchronization skeletons, Numer. Algor. **93**, 731 (2023).
37 S. R. Allmaras, F. T. Johnson, and P. R. Spalart, in Modifications and clarifications for the implementation of the Spalart-Allmaras turbulence model: Proceedings of the 7th International Conference on Computational Fluid Dynamics, Big Island, Hawaii, 2012.
38 P. L. Roe, Approximate Riemann solvers, parameter vectors, and difference schemes, J. Comput. Phys. **43**, 357 (1981).
39 H. Liu, Optimal error estimates of the direct discontinuous Galerkin method for convection-diffusion equations, Math. Comput. **84**, 2263 (2015).
40 Q. Zou, GMRES algorithms over 35 years, Appl. Math. Comput. **445**, 127869 (2023).
41 W. Cao, H. Liu, and Z. Zhang, Superconvergence of the direct discontinuous Galerkin method for convection-diffusion equations, Numer. Meth. Part. D. E. **33**, 290 (2017).
42 T. Poinsot, and S. M. Candel, The influence of differencing and CFL number on implicit time-dependent non-linear calculations, J. Comput. Phys. **62**, 282 (1986).
43 T. Warburton, and T. Hagstrom, Taming the CFL number for discontinuous Galerkin methods on structured meshes, SIAM J. Numer. Anal. **46**, 3151 (2008).
44 S. Joshi, J. Kou, A. Hurtado de Mendoza, K. Puri, C. Hirsch, G. Rubio, and E. Ferrer, Length-scales for efficient CFL conditions in high-order methods with distorted meshes: Application to local-timestepping for p-multigrid, Comput. Fluids **265**, 106011 (2023).
45 Y. H. Tseng, and J. H. Ferziger, A ghost-cell immersed boundary method for flow in complex geometry, J. Comput. Phys. **192**, 593 (2003).
46 NASA, Turbulence Modeling Resource, https://turbmodels.larc.nasa.

gov/.
47 M. Murayama, Y. Yokokawa, H. Ura, K. Nakakita, K. Yamamoto, Y. Ito, T. Takaishi, R. Sakai, K. Shimoda, T. Kato, and T. Homma, in Experimental study of slat noise from 30P30N three-element high-lift airfoil in JAXA Kevlar-wall low-speed wind tunnel: Proceedings of the AIAA/CEAS Aeroacoustics Conference, Atlanta, 2018.

求解任意网格上可压缩Navier-Stokes方程的鲁棒隐式高阶间断伽辽金方法

严佳, 杨小权, 翁培奋

摘要 为了提高高阶方法在模拟复杂结构粘性流动时的鲁棒性和收敛性, 本文提出了一种隐式高阶间断伽辽金(DG)方法. 该种方法在计算稳定性和效率之间实现了良好的平衡, 能够有效地处理复杂流动问题. 具体地, 为了求解线性系统, 发展了精确雅可比矩阵求解方法, 并应用于广义最小残差(GMRES)方法进行预处理和矩阵向量生成. 该方法显著减少了隐式计算中雅可比矩阵的数值误差, 提高了计算的准确性和稳定性. 同时, 通过自适应CFL数增加策略, 进一步提高了隐式方法的计算效率. 此外, 为了提高所提出方法对复杂网格畸变的适应性, 所有的模拟都在参数域中进行. 这种方法显著提高了隐式计算中质量矩阵的可逆性, 从而提高了计算的稳定性. 本文还对影响计算稳定性和效率的各种参数进行了全面分析, 包括CFL数、Krylov子空间大小和GMRES收敛标准. 通过一系列测试算例, 证明了将DG方法、GMRES方法、精确雅可比矩阵计算方法、自适应CFL数和参数域相结合, 能够显著提高计算的鲁棒性、收敛性和计算精度. 这些分析结果为高阶计算中的隐式计算提供了重要的参考价值.