Stabilized seventh-order dissipative compact scheme for two-dimensional Euler equations*

Jia-Xian Qin(秦嘉贤), Ya-Ming Chen(陈亚铭)†, and Xiao-Gang Deng(邓小刚)

College of Aerospace Science and Engineering, National University of Defense Technology, Changsha 410073, *China*

(Received 6 May 2019; revised manuscript received 26 June 2019; published online 17 September 2019)

We derive in this paper a time stable seventh-order dissipative compact finite difference scheme with simultaneous approximation terms (SATs) for solving two-dimensional Euler equations. To stabilize the scheme, the choice of penalty coefficients for SATs is studied in detail. It is demonstrated that the derived scheme is quite suitable for multi-block problems with different spacial steps. The implementation of the scheme for the case with curvilinear grids is also discussed. Numerical experiments show that the proposed scheme is stable and achieves the design seventh-order convergence rate.

Keywords: compact scheme, time stability, simultaneous approximation term, interface treatment

PACS: 47.11.–j, 47.11.Bc, 02.60.Cb **DOI:** 10.1088/1674-1056/ab3f26

1. Introduction

High-order finite difference methods are well suited for simulations of complex physics as they admit high resolution properties and save large amount of computational resources. Typical examples can be found in fluid dynamics.[1,2] Although the derivation of high-order finite difference schemes in the interior of the computational domain is quite straightforward, boundary closures often need special investigation[3–5] to avoid accuracy and stability issues. However, it is not an easy task to construct suitable high-order boundary closures to ensure stability.[3]

In our previous work,[6] we showed that a globally seventh-order scheme is not time stable when boundary conditions are imposed directly with the traditional injection method. To rectify this issue, we employed simultaneous approximation terms (SATs)[7] to impose boundary conditions weakly for a dissipative compact finite difference scheme, resulting in a time stable method with global accuracy of the seventh order. The method was demonstrated to be effective for solving one-dimensional (1D) problems, including linear advection equations and Euler equations. The aim of this paper is to extend the algorithm to solve two-dimensional (2D) Euler equations.

To this end, we need to make some modifications to the scheme since SATs involve some subtle issues in the 2D case. As will be seen in Section 2, around each corner of the 2D computational domain, there exists a small region where the boundary schemes involve boundary values from two different directions, which is different from the 1D case. Therefore, it necessitates to specially discuss the SATs in these regions to ensure stability.

We consider in Section 3 the implementation of the scheme for multi-block problems. In realistic computations the whole computational domain is usually divided into several subdomains with different grid finess to save computational resource considerably (compared with global refinement). For multi-block problems, some interface treatment techniques have to be used.[8–11] Here, the implementation of SATs provides a convenient interface treatment method,[12–14] which has been applied successfully to solve various problems by the summation-by-parts (SBP) community.[15–17]

Since curvilinear grids are often used in practice, we also discuss in Section 4 the implementation of the scheme to this kind of problems by focussing on two relevant examples. The SATs are still used in this case to ensure stability. Finally, conclusions are drawn in Section 5.

2. Extension to the 2D case

Consider the following two-dimensional Euler equations

$$U_t + F(U)_x + G(U)_y = 0, \tag{1}$$

where $U = [\rho, \rho u, \rho v, E]^{\mathrm{T}}$ is the conservative variable, $F = [\rho u, \rho u^2 + p, \rho uv, (E+p)u]^{\mathrm{T}}$ and $G = [\rho v, \rho uv, \rho v^2 + p, (E+p)v]^{\mathrm{T}}$ are the fluxes in different directions. Here the total energy is set to be $E = \rho(u^2+v^2)/2 + p/(\gamma-1)$ with $\gamma = 1.4$ for air.

2.1. Numerical scheme

To extend the seventh-order scheme considered in Ref. [6] (some details can also be found in Appendix A) to solve Eq. (1), it necessitates to consider its quasi-linear form[18]

$$U_t + \boldsymbol{A}(U)U_x + \boldsymbol{B}(U)U_y = 0, \tag{2}$$

where the Jacobian matrices

*Project supported by the National Natural Science Foundation of China (Grant No. 11601517) and the Basic Research Foundation of National University of Defense Technology (Grant No. ZDYYJ-CYJ20140101).

†Corresponding author. E-mail: chenym-08@163.com

© 2019 Chinese Physical Society and IOP Publishing Ltd

$$\boldsymbol{A}(U)=\frac{\partial F}{\partial U}=\begin{pmatrix}0 & 1 & 0 & 0\\ -u^2+\frac{1}{2}(\gamma-1)V^2 & -(\gamma-3)u & -(\gamma-1)v & \gamma-1\\ -uv & v & u & 0\\ u[\frac{1}{2}(\gamma-1)V^2-H] & H-(\gamma-1)u^2 & -(\gamma-1)uv & \gamma u\end{pmatrix}, \tag{3}$$

$$\boldsymbol{B}(U)=\frac{\partial G}{\partial U}=\begin{pmatrix}0 & 0 & 1 & 0\\ uv & v & u & 0\\ -v^2+\frac{1}{2}(\gamma-1)V^2 & -(\gamma-1)u & -(\gamma-3)v & \gamma-1\\ v[\frac{1}{2}(\gamma-1)V^2-H] & -(\gamma-1)uv & H-(\gamma-1)v^2 & \gamma v\end{pmatrix}. \tag{4}$$

Here $V^2=u^2+v^2$, and $H=V^2/2+\gamma p/(\gamma-1)\rho$ is the specific enthalpy. In addition, the Jacobian matrices can be diagonalized as $\boldsymbol{\Lambda}_A=\boldsymbol{K}^{-1}\boldsymbol{A}\boldsymbol{K}=\mathrm{diag}\{u-a,u,u,u+a\}$ and $\boldsymbol{\Lambda}_B=\boldsymbol{P}^{-1}\boldsymbol{B}\boldsymbol{P}=\mathrm{diag}\{v-a,v,v,v+a\}$, where $a=\sqrt{\gamma p/\rho}$ represents the sound speed, $\boldsymbol{K}$ and $\boldsymbol{P}$ read explicitly as

$$\boldsymbol{K}=\begin{pmatrix}1 & 1 & 0 & 1\\ u-a & u & 0 & u+a\\ v & v & 1 & v\\ H-ua & \frac{1}{2}V^2 & v & H+ua\end{pmatrix},$$
$$\boldsymbol{P}=\begin{pmatrix}1 & 1 & 0 & 1\\ u & u & 1 & u\\ v-a & v & 0 & v+a\\ H-va & \frac{1}{2}V^2 & u & H+va\end{pmatrix}. \tag{5}$$

For a computational domain $(x,y)\in[x_w,x_e]\times[y_s,y_n]$, we denote the discrete coordinates by $x_i=x_w+(i-1)h_x$ with $1\le i\le N$ and $y_j=y_s+(j-1)h_y$ with $1\le j\le M$, where $h_x=(x_e-x_w)/(N-1)$ and $h_y=(y_n-y_s)/(M-1)$ are the spacial steps in different directions. Let $U_{i,j}$ and $(F_x)_{i,j}$ be the approximations to $U(x_i,y_j)$ and $F(U(x_i,y_j))_x$, respectively, and similar notations for other fluxes and matrices. Then the interior semi-discrete scheme for Eq. (1) can be written as

$$\begin{aligned}&\frac{\mathrm{d}}{\mathrm{d}t}U_{i,j}+(F_x)_{i,j}+(G_y)_{i,j}=0,\\ &5\le i\le N-4,\quad 5\le j\le M-4,\end{aligned} \tag{6}$$

where the expressions for $(F_x)_{i,j}$ and $(G_y)_{i,j}$ are presented in Appendix A (see also Ref. [6]). For other points near boundaries, boundary values are involved, which are imposed weakly by using SATs. By introducing the notation

$$S_{i,j}=\frac{\mathrm{d}}{\mathrm{d}t}U_{i,j}+(F_x)_{i,j}+(G_y)_{i,j},$$

the approximations at the four corner regions read

$$\begin{aligned}S_{i,j}=\;&(-1)^i\frac{\tau_i}{h_x}\bar{\boldsymbol{A}}^+_{i,j}(U_{1,j}-C^w_j)+(-1)^j\frac{\tau_j}{h_y}\bar{\boldsymbol{B}}^+_{i,j}(U_{i,1}-C^s_i),\\ &1\le i\le 4,1\le j\le 4,\end{aligned} \tag{7}$$

$$\begin{aligned}S_{i,j}=\;&(-1)^{N-i}\frac{\tau_{N-i+1}}{h_x}\bar{\boldsymbol{A}}^-_{i,j}(U_{N,j}-C^e_j)\\ &+(-1)^j\frac{\tau_j}{h_y}\bar{\boldsymbol{B}}^+_{i,j}(U_{i,1}-C^s_i),\\ &N-3\le i\le N,1\le j\le 4,\end{aligned} \tag{8}$$

$$\begin{aligned}S_{i,j}=\;&(-1)^i\frac{\tau_i}{h_x}\bar{\boldsymbol{A}}^+_{i,j}(U_{1,j}-C^w_j)\\ &+(-1)^{M-j}\frac{\tau_{M-j+1}}{h_y}\bar{\boldsymbol{B}}^-_{i,j}(U_{i,M}-C^n_i),\\ &1\le i\le 4,M-3\le j\le M,\end{aligned} \tag{9}$$

$$\begin{aligned}S_{i,j}=\;&(-1)^{N-i}\frac{\tau_{N-i+1}}{h_x}\bar{\boldsymbol{A}}^-_{i,j}(U_{N,j}-C^e_j)\\ &+(-1)^{M-j}\frac{\tau_{M-j+1}}{h_y}\bar{\boldsymbol{B}}^-_{i,j}(U_{i,M}-C^n_i),\\ &N-3\le i\le N,M-3\le j\le M.\end{aligned} \tag{10}$$

For other points, the following semi-discrete approximations are used:

$$\begin{aligned}S_{i,j}=\;&(-1)^i\frac{\tau_i}{h_x}\bar{\boldsymbol{A}}^+_{i,j}(U_{1,j}-C^w_j),\\ &1\le i\le 4,4\le j\le M-3,\end{aligned} \tag{11}$$

$$\begin{aligned}S_{i,j}=\;&(-1)^{N-i}\frac{\tau_{N-i+1}}{h_x}\bar{\boldsymbol{A}}^-_{i,j}(U_{N,j}-C^e_j),\\ &N-3\le i\le N,4\le j\le M-3,\end{aligned} \tag{12}$$

$$\begin{aligned}S_{i,j}=\;&(-1)^j\frac{\tau_j}{h_y}\bar{\boldsymbol{B}}^+_{i,j}(U_{i,1}-C^s_i),\\ &4\le i\le N-3,1\le j\le 4,\end{aligned} \tag{13}$$

$$\begin{aligned}S_{i,j}=\;&(-1)^{M-j}\frac{\tau_{M-j+1}}{h_y}\bar{\boldsymbol{B}}^-_{i,j}(U_{i,M}-C^n_i),\\ &4\le i\le N-3,M-3\le j\le M.\end{aligned} \tag{14}$$

Here τ_k $(1\le k\le 4)$ are penalty coefficients that can be tuned to stabilize the scheme, C^w_j, C^e_j, C^s_i, and C^n_i stand for the well-posed west, east, south, and north boundary conditions, respectively. In addition, $\bar{\boldsymbol{A}}^\pm_{i,j}=\bar{\boldsymbol{K}}_{i,j}\bar{\boldsymbol{\Lambda}}^\pm_{A,i,j}\bar{\boldsymbol{K}}^{-1}_{i,j}$ with $\bar{\boldsymbol{\Lambda}}^\pm_{A,i,j}=(\bar{\boldsymbol{\Lambda}}_{A,i,j}\pm|\bar{\boldsymbol{\Lambda}}_{A,i,j}|)/2$, where the notation $\bar{\boldsymbol{A}}$ represents the frozen value of $\boldsymbol{A}$ at the last time step. Similar notations also apply for $\bar{\boldsymbol{B}}^\pm_{i,j}$. It is observed that the added SATs $S_{i,j}$ in Eqs. (7)–(14) vanish when the exact solution is switched into the scheme, meaning that the implementation of SAT method does not contribute to the truncation error and thus does not affect the convergence rate of the original scheme.

Intuitively, one may simply choose the penalty coefficients to be same values as the one-dimensional case, *i.e.*, $\tau_1=\tau_2=\tau_3=\tau_4=5$.[6] To test this choice, we take as an example the vortex model from Ref. [19], whose solution is given by

$$\rho(x,y,t)=\left[1-\varepsilon^2\frac{\gamma-1}{8\pi^2}\exp(f(x,y,t))\right]^{1/(\gamma-1)}, \tag{15}$$

$$u(x,y,t)=Ma_\infty-\varepsilon\frac{y-y_0}{2\pi}\exp\left(\frac{f(x,y,t)}{2}\right), \tag{16}$$

$$v(x,y,t)=\varepsilon\frac{x-x_0-Ma_\infty t}{2\pi}\exp\left(\frac{f(x,y,t)}{2}\right), \tag{17}$$

$$p(x,y,t) = \rho^{\gamma}/\gamma, \tag{18}$$

where $f(x,y,t) = 1 - [(x - x_0 - Ma_\infty t)^2 + (y - y_0)^2]$. Here the parameters $(x_0, y_0) = (5,0)$, $Ma_\infty = 0.5$, $\varepsilon = 1.0$, and $\gamma = 1.4$ are used. The computational domain is set to be $(x,y) \in [0,10] \times [-5,5]$ and the time step $\Delta t = 0.1 h_x h_y$ is chosen to implement a traditional fourth-order Runge–Kutta scheme. If not stated otherwise, this time scheme will be used for other numerical examples presented in this paper. For simplicity, exact solution values are adopted at the boundaries. One can observe from Fig. 1 that at $t = 4.2$ the density solution near the right corners starts to deteriorate, while at $t = 6.3$ the vortex is almost broken. This phenomenon indicates that some modifications for the scheme should be made to dissolve the non-physical error emerging from the corner regions, where the SATs from two boundaries overlap; see Fig. 2(a).

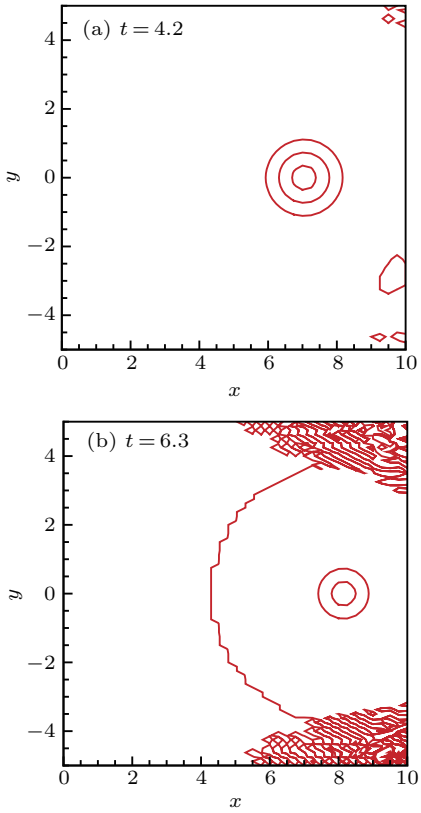


Fig. 1. Density contours of the vortex obtained by the scheme of Eqs. (6)–(14) with penalty coefficients $\tau_1 = \tau_2 = \tau_3 = \tau_4 = 5$. Here $N \times M = 41 \times 81$, the time step $\Delta t = 0.1 h_{x_1} h_{y_1}$ is chosen to implement the fourth-order Runge–Kutta scheme. (a) At the moment $t = 4.2$. (b) At the moment $t = 6.3$.

To show how to improve the scheme, we take as an example the solution point (x_1, y_{M-1}) at the west–north corner; see Fig. 2(b). The semi-discrete approximation (9) at this point reads

$$\begin{aligned} &\frac{\mathrm{d}}{\mathrm{d}t} U_{1,M-1} + (F_x)_{1,M-1} + (G_y)_{1,M-1} \\ &= -\frac{\tau_1}{h_x} \bar{A}^{+}_{1,M-1} (U_{1,M-1} - C^{w}_{M-1}) \\ &\quad - \frac{\tau_2}{h_y} \bar{B}^{-}_{1,M-1} (U_{1,M} - C^{n}_{1}). \end{aligned} \tag{19}$$

Here the first penalty term imposes weakly the west boundary condition, whereas the second one imposes the north boundary condition. It is noted that the solution point (x_1, y_{M-1}) resides on the west boundary, which means that the second penalty term is not so important here. In this regard, we adjust the coefficient of the first penalty coefficient τ_1 to weaken the effect of the second one, while other coefficients remain unchanged. The same reasoning applies for other similar points. In this paper, we choose the penalty coefficients to be

$$\tau_1 = 20, \quad \tau_2 = \tau_3 = \tau_4 = 5 \tag{20}$$

for the scheme of Eqs. (6)–(14), although the choice may be optimized further.

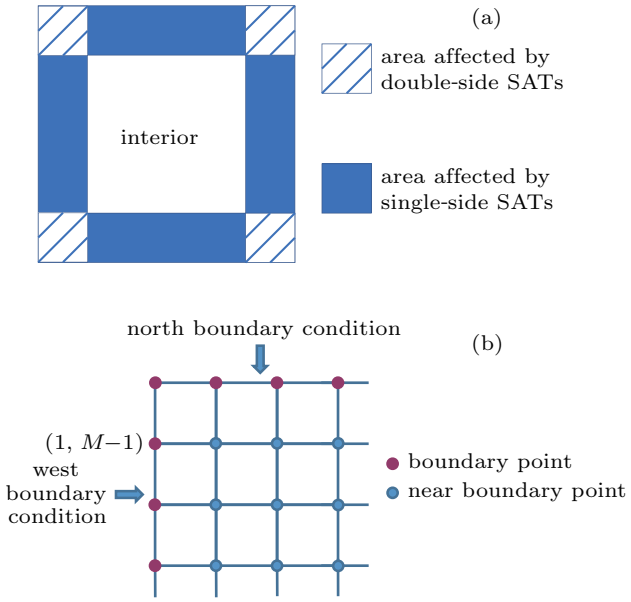


Fig. 2. (a) Illustration of the computational domain. (b) The north–west corner area, where the solution points are affected by SATs from both boundaries.

2.2. Numerical example

Using the scheme of Eqs. (6)–(14) with the penalty coefficients (20), we recalculate the vortex solution of Eqs. (15)–(18). The new numerical results are presented in Fig. 3, showing good preservation of the vortex. For accuracy test, we introduce the following L^∞ norm and L^2 norm:

$$\begin{aligned} ||e||_\infty &= \max_{i,j} |U_{i,j} - U(x_i, y_j, t)|, \\ ||e||_2 &= \left(\sum_{i=1}^{N} \sum_{j=1}^{M} (U_{i,j} - U(x_i, y_j, t))^2 h_x h_y \right)^{1/2}. \end{aligned} \tag{21}$$

The convergence rate is measured by

$$-\log_{10}(e_1/e_2)/\log_{10}(N_1/N_2),$$

where e_1 and e_2 are the corresponding errors for different numbers of grid cells N_1 and N_2, respectively. The results shown in Table 1 indicate that the expected seventh-order convergence rate is achieved approximately.

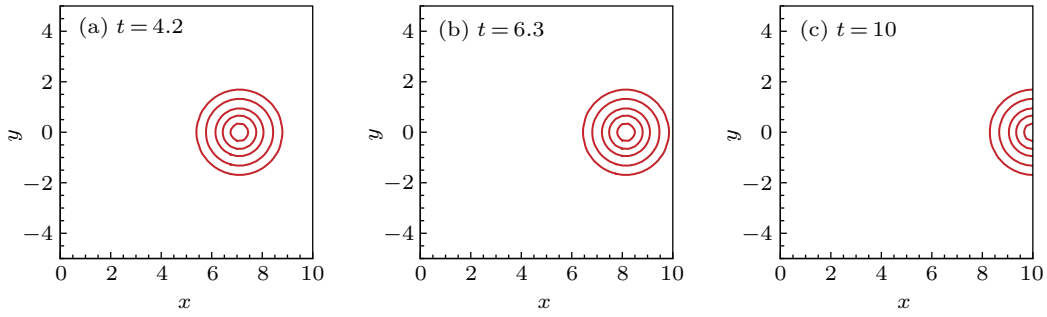


Fig. 3. Density contours obtained by the scheme of Eqs. (6)–(14) with the penalty coefficients (20) on a 41×81 grid. Here the time step $\Delta t = 0.1 h_{x_1} h_{y_1}$ is chosen to implement the fourth-order Runge–Kutta scheme. (a) At the moment $t = 4.2$. (b) At the moment $t = 6.3$. (c) At the moment $t = 10$.

Table 1. Numerical test for the vortex solution of Eqs. (15)–(18) at $t = 10$. Here the time step $\Delta t = 0.1 h_x h_y$ is chosen to implement the fourth-order Runge–Kutta scheme.

$N \times M$	L^2 error	Order	L^∞ error	Order
41×81	9.51×10^{-7}	–	1.19×10^{-5}	–
61×121	6.02×10^{-8}	6.81	7.93×10^{-7}	6.69
81×161	8.51×10^{-9}	6.80	9.66×10^{-8}	7.32

3. Multi-block problems

In this section we intend to show that the developed scheme with SATs is well suited for multi-block problems, which are often used for complex configurations.

3.1. Interface treatment

For simplicity, the computational domain $[x_w, x_e] \times [y_s, y_n]$ is only divided into two subdomains with an interface situated at $x_k = (x_e - x_w)/2$; see Fig. 4. Denote the conservative variables in the left and right subdomains by $U_{i,j}$ with $1 \le i \le N_1, 1 \le j \le M$ and $V_{i,j}$ with $1 \le i \le N_2, 1 \le j \le M$, respectively. In addition, assume that $V_{1,j}$ are the east boundary values for the left subdomain, and $U_{N_1,j}$ are the west boundary values for the right. Then the problem can be solved by using the scheme of Eqs. (6)–(14) separately for each subdomain.

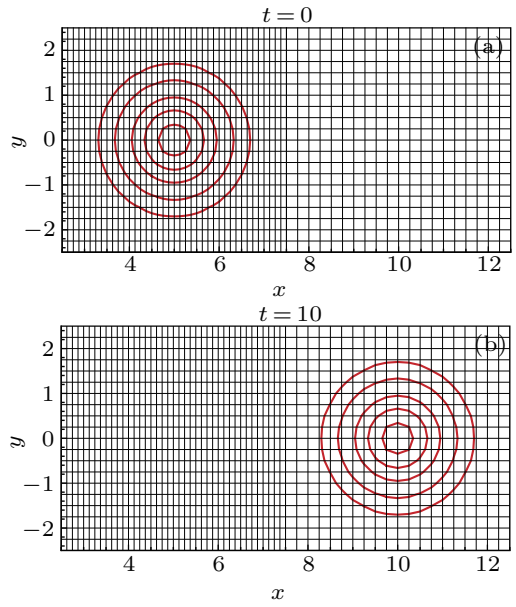


Fig. 4. Density contours of the vortex of Eqs. (15)–(18) on the grid consists of two subdomains with 41×21 and 21×21 points. Here the time step $\Delta t = 0.1 h_{x_1} h_{y_1}$ is chosen to implement the fourth-order Runge–Kutta scheme. (a) At the moment $t = 0$. (b) At the moment $t = 10$.

To verify the scheme, we calculate the vortex convection problem of Eqs. (15)–(18) on the computational domain $(x, y) \in [2.5, 12.5] \times [-2.5, 2.5]$ with the split line $x = 7.5$. The grid ratio is set to be $2 : 1$ in the x direction. Figure 4 depicts that the vortex propagates successfully through the interface. One can also see from Table 2 that the convergence rates in both the two subdomains are well preserved.

Table 2. Numerical test for the vortex solution of Eqs. (15)–(18) on two different subdomains with the grid ratio $2 : 1$. Here the time step $\Delta t = h_{x_1} h_{y_1}$ is chosen to implement the fourth-order Runge–Kutta scheme.

$N \times M$	L^2 error	Order	L^∞ error	Order
41×21	1.19×10^{-6}	–	2.79×10^{-6}	–
61×41	7.97×10^{-8}	6.66	1.71×10^{-7}	6.89
81×41	1.10×10^{-8}	6.89	2.34×10^{-8}	6.90

3.2. Junction point

Next we demonstrate that by exchanging information through penalty terms, the interface technique can naturally handle grid partition with junction points. This time we divide the computational domain into four subdomains with two interfaces situated at $x_k = (x_e - x_w)/2$ and $y_l = (y_n - y_s)/2$, respectively. Here we still apply directly the scheme of Eqs. (6)–(14) to each subdomain with the boundary values at interfaces setting to be those of corresponding adjacent points; see Fig. 5.

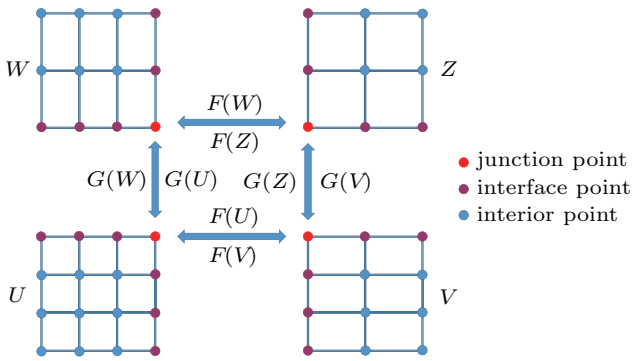


Fig. 5. Illustration of information permutation at the junction point.

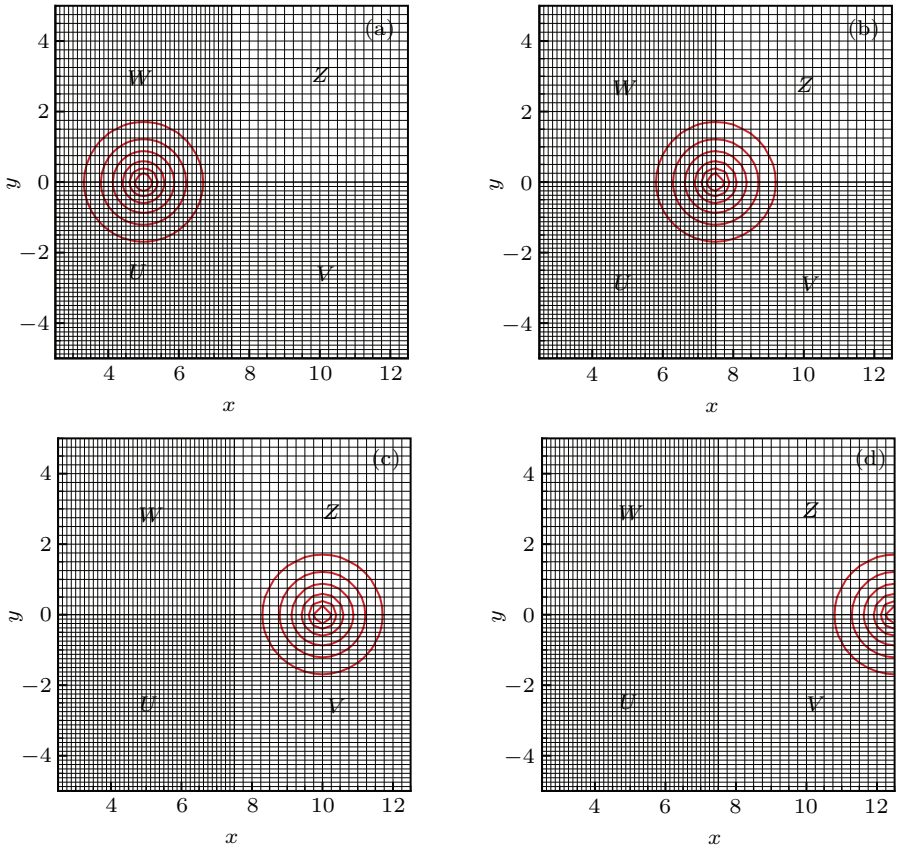


Fig. 6. Density contours on the grid consists of four subdomains with 41×41 (U), 41×21 (V), 21×41 (W), and 21×21 (Z) points. Here the time step $\Delta t = 0.1h_{x_1}h_{y_1}$ is chosen to implement the fourth-order Runge–Kutta scheme. (a) At the moment $t=0$. (b) At the moment $t=5$. (c) At the moment $t=10$. (d) At the moment $t=15$.

Table 3. Numerical test for the vortex solution of Eqs. (15)–(18) at time $t=10$ on four different subdomains with the grid ratio set to be the same as that presented in Fig. 6. For brevity only the number of solution points in the subdomain U are presented in the table. Here the time step $\Delta t = 0.1h_{x_1}h_{y_1}$ is chosen to implement the fourth-order Runge–Kutta scheme.

	Subdomain U		Subdomain V		Subdomain W		Subdomain Z	
$N_1\times M_1$	L^2 error	Order	L^2 error	Order	L^2 error	Order	L^2 error	Order
61×61	5.75×10^{-8}	–	1.60×10^{-7}	–	7.89×10^{-8}	–	2.54×10^{-7}	–
91×91	3.49×10^{-9}	6.91	1.03×10^{-8}	6.78	5.06×10^{-9}	6.77	1.61×10^{-8}	6.80
121×121	4.65×10^{-10}	7.00	1.33×10^{-9}	7.10	6.79×10^{-10}	6.98	2.04×10^{-9}	7.20

Once again we consider the vortex convection problem of Eqs. (15)–(18) to verify the scheme. Here the computational domain $(x,y)\in[2.5,12.5]\times[-5,5]$ is chosen with interfaces situated at $x=10$ and $y=0$. It is observed from Fig. 6 that the vortex propagates through the interfaces smoothly. In addition, the accuracy test presented in Table 3 shows that the scheme achieves its design order of accuracy approximately.

4. Discussions on curvilinear grids

In this section, we study how to implement the numerical scheme for problems with curvilinear grids, which are often needed in practice for complex configurations. In this case, we need to consider the problem in the computational coordinates (ξ,η), where the Euler equations (1) read

$$JU_t+\widehat{F}_\xi+\widehat{G}_\eta=0, \tag{22}$$

where the Jacobian $J=x_\xi y_\eta-x_\eta y_\xi$, $\widehat{F}=y_\eta F-x_\eta G$, $\widehat{G}=-y_\xi F+x_\xi G$. Here, we apply the difference scheme presented in Appendix A to calculate numerically the metrics x_ξ, x_η, y_ξ, y_η, and the Jacobian J. To ensure stability, it still necessitates to use SATs similarly to the case for Eq. (1) on Cartesian grids; see *e.g.*, Eq. (19). For convenience of the reader, some necessary formulae of the Jacobian matrices of the fluxes $\widehat{F}$ and $\widehat{G}$ are presented in Appendix B to implement the similar scheme as Eqs. (6)–(14).

4.1. Stationary model

The first example is a model governed by the following Euler equations with a source term

$$U_t+F_x+G_y=S, \tag{23}$$

where the source term is determined analytically by $S=F_x+G_y$ so that the problem admits the following stationary

solution[20]

$$\rho = 0.5\left[\sin(x^2+y^2)+1.5\right], \tag{24}$$

$$u = \sin(x^2+y^2)+0.5, \tag{25}$$

$$v = 0.1\left[\cos(x^2+y^2)+0.5\right], \tag{26}$$

$$E = 0.5\left[\cos(x^2+y^2)+1.5\right]. \tag{27}$$

The numerical test for the stationary model of Eqs. (23)–(27) is shown in the following Table 4.

Table 4. Numerical test for the stationary model of Eqs. (23)–(27). Here the time step $\Delta t = 0.01h_x$ is chosen to implement the implicit Euler scheme.

$N\times M$	L^2 error	Order	L^∞ error	Order
41×41	2.46×10^{-7}	–	6.15×10^{-7}	–
61×61	2.54×10^{-8}	5.60	7.07×10^{-8}	5.34
81×81	3.43×10^{-9}	6.96	1.00×10^{-8}	6.80
121×121	1.86×10^{-10}	7.18	6.44×10^{-10}	6.77

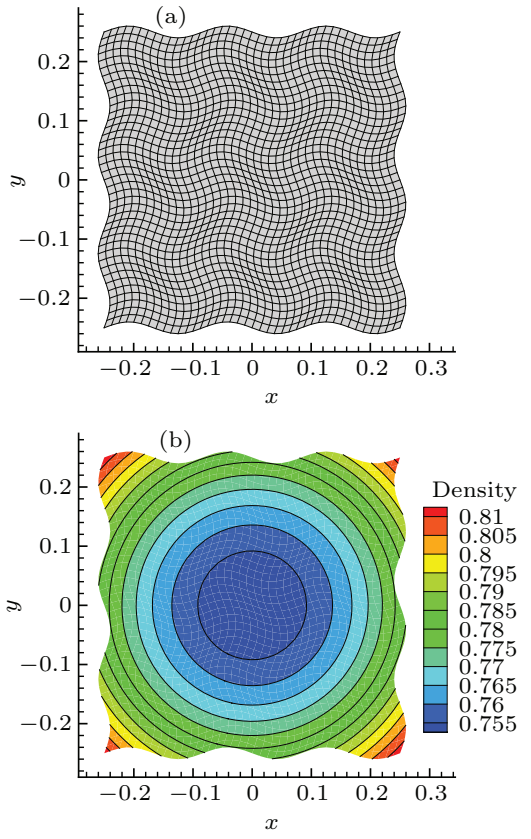


Fig. 7. (a) Illustration of a 41×41 curvilinear grid. (b) Final density contour obtained on the 41×41 grid.

The curvilinear grid[21] considered here (see Fig. 7(a)) is generated from the standard computational domain $(\xi,\eta)\in[0,1]\times[0,1]$ by

$$x(\xi,\eta) = -0.5L + L[\xi + \mu\sin(\lambda\pi\eta)], \tag{28}$$

$$y(\xi,\eta) = -0.5L + L[\eta + \mu\sin(\lambda\pi\xi)], \tag{29}$$

where the parameters $\mu = 0.02$, $\lambda = 6$, and $L = 0.5$ are chosen. The implicit Euler scheme is implemented for time-marching until the residue reaches machine zero (see Fig. 7(b) for the final solution).

4.2. Nozzle flow

Consider a channel flow[22,23] governed by the Euler equations (1). The physical domain (Fig. 8(a)) is bounded between $x = -0.75$ and $x = 0.75$ in the x direction, and is bounded by

$$y_1(x) = 0.05 - 0.1\times\left(\frac{20}{9}\right)^2 x^2 + 0.05\times\left(\frac{20}{9}\right)^4 x^4, \tag{30}$$

$$y_2(x) = 0.95 + 0.1\times\left(\frac{20}{9}\right)^2 x^2 - 0.05\times\left(\frac{20}{9}\right)^4 x^4 \tag{31}$$

in the y direction. The grid is produced analytically by the expressions

$$x = \xi, \tag{32}$$

$$y = (1-\eta)y_1(\xi) + \eta y_2(\xi), \tag{33}$$

with ξ and η being partitioned equally.

The initial flow field is set to be the free stream with $Ma_\infty = 0.3$, $\rho_\infty = 1$, $p_\infty = 1$, $u_\infty = Ma_\infty\cdot a_\infty$, $v_\infty = 0$, where $a_\infty = \sqrt{\gamma p_\infty/\rho_\infty}$ represents the sound speed. The left boundary is set to be subsonic inflow while the right is subsonic outflow. Both the top and the bottom are set to be slip-wall boundary conditions. How to impose the slip-wall boundary condition weakly is explained in Appendix C.

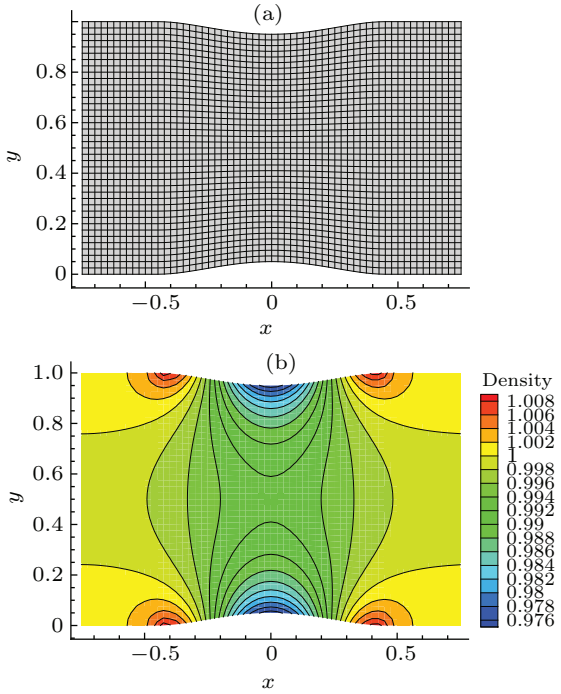


Fig. 8. A grid of the channel with 61×41 solution points and the corresponding numerical result of the density. (a) The 61×41 mesh. (b) Final flow field.

We run the calculation on a grid with 61×41 solution points until the residue reaches the machine zero. It can be

observed from Fig. 8(b) that the obtained density contour is smooth, showing the effectiveness of the scheme. Similar results can also be obtained on other grids with different finess. As the focus here is just to test whether the scheme is applicable for the case with curvilinear grids and wall boundary conditions, the convergence rate will not be discussed, which depends also on the smoothness of grids and the implementation of boundary conditions.

In addition, to compare the results obtained by the modified scheme and the scheme with the same penalty coefficients as Ref. [6], we show in Fig. 9 the density contours and velocity vectors near the bump at $t = 0.054$. It is observed that while the unmodified scheme starts to break up, the modified one still preserves well, demonstrating again that the modified coefficients are necessary to ensure stability.

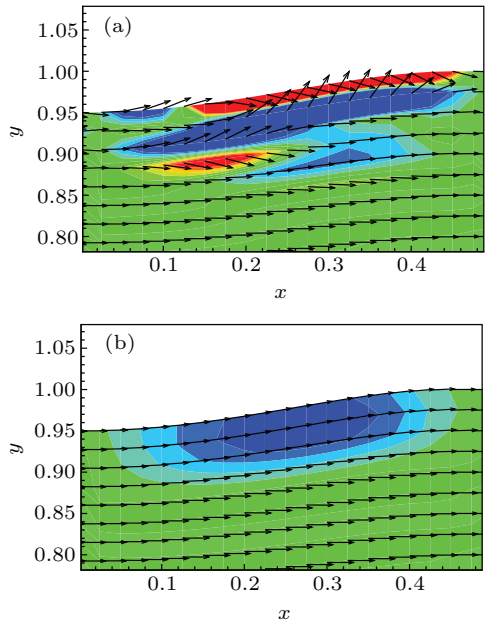


Fig. 9. Density solutions and velocity vectors near the bump obtained on the 61×41 grid at $t = 0.054$. (a) The unmodified result. (b) The modified result.

5. Conclusion

In this paper, we generalized the globally seventh-order dissipative compact scheme with SATs[6] to the 2D Euler equations. The choice of penalty coefficients for SATs was reconsidered to stabilize the scheme. It was shown that the scheme with SATs is very convenient for dealing with multi-block problems with conformal grids. In addition, the implementation of the scheme for the case with curvilinear grids was also discussed, including the slip-wall boundary condition. Various numerical experiments were performed to verify the proposed scheme. The extension to Navier–Stokes equations may be considered in further work.

Appendix A: Spacial discretizations

Here we revisit briefly the spacial schemes presented in Ref. [6]. For the one-dimensional conservation law

$$U_t + F_x = 0, \quad x_l \leqslant x \leqslant x_r, \tag{A1}$$

partition the computational domain $[x_l, x_r]$ by using solution points $x_j = x_l + (j-1)h$ $(1 \leqslant j \leqslant N)$ and flux points $x_{j+1/2} = x_l + (j-1/2)h$ $(1 \leqslant j \leqslant N-1)$, where N is the number of solution points, and $h = (x_r - x_l)/(N-1)$ is the spacial step. Let U_j and F'_j be approximations to $U(x_j,t)$ and $F(U(x_j,t))_x$, respectively. Notations for other variables and matrices introduced in Section 2 are still adopted. The exact left and right boundary values are given by

$$U(x_l,t) = C^l(t), \quad U(x_r,t) = C^r(t). \tag{A2}$$

Then the derivative of flux can be computed by the following formulae: for $5 \leq j \leq N-4$ (*i.e.*, the interior points),

$$F'_j = \frac{1}{h}\left[\frac{256}{175}(F_{j+1/2} - F_{j-1/2}) - \frac{1}{4}(F_{j+1} - F_{j-1}) + \frac{1}{100}(F_{j+2} - F_{j-2}) - \frac{1}{2100}(F_{j+3} - F_{j-3})\right]. \tag{A3}$$

For near left boundary points we use the following difference schemes:

$$F'_1 = \frac{1}{h}\left(12F_{3/2} + \frac{40}{3}F_{5/2} + \frac{12}{5}F_{7/2} - \frac{49}{10}F_1 - 15F_2 - \frac{15}{2}F_3 - \frac{1}{3}F_4\right), \tag{A4}$$

$$F'_2 = \frac{1}{h}\left(-\frac{4}{5}F_{3/2} + \frac{8}{3}F_{5/2} + \frac{4}{15}F_{7/2} + \frac{1}{15}F_1 - \frac{7}{6}F_2 - F_3 - \frac{1}{30}F_4\right), \tag{A5}$$

$$F'_3 = \frac{1}{h}\left(-\frac{64}{45}F_{5/2} + \frac{64}{45}F_{7/2} - \frac{1}{180}F_1 + \frac{2}{9}F_2 - \frac{2}{9}F_4 + \frac{1}{180}F_5\right), \tag{A6}$$

$$F'_4 = \frac{1}{h}\left(-\frac{256}{175}F_{7/2} + \frac{256}{175}F_{9/2} + \frac{1}{2100}F_1 - \frac{1}{100}F_2 + \frac{1}{4}F_3 - \frac{1}{4}F_5 + \frac{1}{100}F_6 - \frac{1}{2100}F_7\right), \tag{A7}$$

and the schemes for the right can be obtained by symmetry.

While the values F_j involved in the above difference scheme can be computed directly by $F_j = F(U_j)$, the values $F_{j+1/2} = \tilde{F}(U^L_{j+1/2}, U^R_{j+1/2})$ are calculated by any approximate Riemann solver (in this paper Roe's Riemann solver is adopted). Here $U^{L,R}_{j+1/2}$ stands for the left and the right values obtained by using the seventh-order dissipative compact interpolation scheme[1]: For the interior flux points $\xi_{j+1/2}$ $(3 \leq j \leq N-3)$,

$$\frac{4}{15}(1-\alpha)U_{j-1/2}+U_{j+1/2}+\frac{4}{15}(1+\alpha)U_{j+3/2}=\frac{25}{32}(U_{j+1}+U_j)+\frac{5}{64}(U_{j+2}+U_{j-1})-\frac{1}{448}(U_{j+3}+U_{j-2})$$
$$+\alpha\left[\frac{25}{64}(U_{j+1}-U_j)+\frac{15}{128}(U_{j+2}-U_{j-1})-\frac{5}{896}(U_{j+3}-U_{j-2})\right], \tag{A8}$$

where the dissipation constant $\alpha=-0.3$ is used for the left upwind interpolation, and by setting $\alpha=0.3$ the interpolation becomes right upwind automatically. Due to the limitation of stencil length, the interpolations at near boundary flux points become

$$U_{3/2}=\frac{231}{1024}U_1+\frac{693}{512}U_2-\frac{1155}{1024}U_3+\frac{231}{256}U_4\frac{495}{1024}U_5+\frac{77}{512}U_6-\frac{21}{1024}U_7, \tag{A9}$$
$$U_{5/2}=-\frac{21}{1024}U_1+\frac{189}{512}U_2+\frac{945}{1024}U_3-\frac{105}{256}U_4+\frac{189}{1024}U_5-\frac{27}{512}U_6+\frac{7}{1024}U_7, \tag{A10}$$
$$U_{N-3/2}=\frac{7}{1024}U_{N-6}-\frac{27}{512}U_{N-5}+\frac{189}{1024}U_{N-4}-\frac{105}{256}U_{N-3}+\frac{945}{1024}U_{N-2}+\frac{189}{512}U_{N-1}-\frac{21}{1024}U_N, \tag{A11}$$
$$U_{N-1/2}=-\frac{21}{1024}U_{N-6}+\frac{77}{512}U_{N-5}-\frac{495}{1024}U_{N-4}+\frac{231}{256}U_{N-3}-\frac{1155}{1024}U_{N-2}+\frac{693}{512}U_{N-1}+\frac{231}{1024}U_N. \tag{A12}$$

Appendix B: Jacobian matrices of fluxes in Eq. (22)

The Jacobian matrices of $\widehat{F}$ and $\widehat{G}$ in Eq. (22) are

$$\widehat{\boldsymbol{A}}=\frac{\partial\widehat{F}}{\partial U}=J\begin{pmatrix}0 & \xi_x & \xi_y & 0\\ \xi_x\Phi-u\theta_1 & \theta_1-\xi_x(\gamma-2)u & \xi_y u-\xi_x(\gamma-1)v & \xi_x(\gamma-1)\\ \xi_y\Phi-v\theta_1 & \xi_x v-\xi_y(\gamma-1)u & \theta_1-\xi_y(\gamma-2)v & \xi_y(\gamma-1)\\ \theta_1(2\Phi-\gamma e) & \xi_x(\gamma e-\Phi)-(\gamma-1)u\theta_1 & \xi_y(\gamma e-\Phi)-(\gamma-1)v\theta_1 & \gamma\theta_1\end{pmatrix}, \tag{B1}$$

$$\widehat{\boldsymbol{B}}=\frac{\partial\widehat{G}}{\partial U}=J\begin{pmatrix}0 & \eta_x & \eta_y & 0\\ \eta_x\Phi-u\theta_2 & \theta_2-\eta_x(\gamma-2)u & \eta_y u-\eta_x(\gamma-1)v & \eta_x(\gamma-1)\\ \eta_y\Phi-v\theta_2 & \eta_x v-\eta_y(\gamma-1)u & \theta_2-\eta_y(\gamma-2)v & \eta_y(\gamma-1)\\ \theta_2(2\Phi-\gamma e) & \eta_x(\gamma e-\Phi)-(\gamma-1)u\theta_2 & \eta_y(\gamma e-\Phi)-(\gamma-1)v\theta_2 & \gamma\theta_2\end{pmatrix}, \tag{B2}$$

where J is defined in Eq. (22), $\Phi=(\gamma-1)(u^2+v^2)/2$, $\theta_1=\xi_x u+\xi_y v$, $\theta_2=\eta_x u+\eta_y v$.

Diagonalizing $\widehat{\boldsymbol{A}}$ yields $\widehat{\boldsymbol{\Lambda}}_A=\widehat{\boldsymbol{K}}^{-1}\widehat{\boldsymbol{A}}\widehat{\boldsymbol{K}}=J\mathrm{diag}[\theta_1,\theta_1,\theta_1-k_1a,\theta_1+k_1a]$, where $k_1=\sqrt{\xi_x^2+\xi_y^2}$, $a=\sqrt{\gamma p/\rho}$,

$$\widehat{\boldsymbol{K}}=\begin{pmatrix}1 & 0 & \chi_1 & \chi_1\\ u & \bar{\xi}_y & \chi_1(u-a\bar{\xi}_x) & \chi_1(u+a\bar{\xi}_x)\\ v & -\bar{\xi}_x & \chi_1(v-a\bar{\xi}_y) & \chi_1(v+a\bar{\xi}_y)\\ \frac{u^2+v^2}{2} & \bar{\xi}_y u-\bar{\xi}_x v & \chi_1(H-a\bar{\theta}_1) & \chi_1(H+a\bar{\theta}_1)\end{pmatrix}, \tag{B3}$$

$$\widehat{\boldsymbol{K}}^{-1}=\begin{pmatrix}1-\Phi/a^2 & (\gamma-1)u/a^2 & (\gamma-1)v/a^2 & -(\gamma-1)/a^2\\ \bar{\xi}_x v-\bar{\xi}_y u & \bar{\xi}_y & -\bar{\xi}_x & 0\\ \chi_2(\Phi+a\bar{\theta}_1) & -\chi_2[\bar{\xi}_x a+(\gamma-1)u] & -\chi_2[\bar{\xi}_y a+(\gamma-1)v] & \chi_2(\gamma-1)\\ \chi_2(\Phi-a\bar{\theta}_1) & \chi_2[\bar{\xi}_x a-(\gamma-1)u] & \chi_2[\bar{\xi}_y a-(\gamma-1)v] & \chi_2(\gamma-1)\end{pmatrix} \tag{B4}$$

with $\chi_1=\rho/(\sqrt{2}a)$, $\chi_2=1/(\sqrt{2}\rho a)$, $\bar{\xi}_x=\xi_x/k_1$, $\bar{\xi}_y=\xi_y/k_1$ and $\bar{\theta}_1=\theta_1/k_1$.

Similarly, we have $\widehat{\boldsymbol{\Lambda}}_B=\widehat{\boldsymbol{P}}^{-1}\widehat{\boldsymbol{B}}\widehat{\boldsymbol{P}}=J\mathrm{diag}[\theta_2,\theta_2,\theta_2-k_2a,\theta_2+k_2a]$, where $k_2=\sqrt{\eta_x^2+\eta_y^2}$,

$$\widehat{\boldsymbol{P}}=\begin{pmatrix}1 & 0 & \chi_1 & \chi_1\\ u & \bar{\eta}_y & \chi_1(u-a\bar{\eta}_x) & \chi_1(u+a\bar{\eta}_x)\\ v & -\bar{\eta}_x & \chi_1(v-a\bar{\eta}_y) & \chi_1(v+a\bar{\eta}_y)\\ (u^2+v^2)/2 & \bar{\eta}_y u-\bar{\eta}_x v & \chi_1(H-a\bar{\theta}_1) & \chi_1(H+a\bar{\theta}_1)\end{pmatrix}, \tag{B5}$$

$$\widehat{\boldsymbol{P}}^{-1}=\begin{pmatrix}1-\Phi/a^2 & (\gamma-1)u/a^2 & (\gamma-1)v/a^2 & -(\gamma-1)/a^2\\ \bar{\eta}_x v-\bar{\eta}_y u & \bar{\eta}_y & -\bar{\eta}_x & 0\\ \chi_2(\Phi+a\bar{\theta}_2) & -\chi_2[\bar{\eta}_x a+(\gamma-1)u] & -\chi_2[\bar{\eta}_y a+(\gamma-1)v] & \chi_2(\gamma-1)\\ \chi_2(\Phi-a\bar{\theta}_2) & \chi_2[\bar{\eta}_x a-(\gamma-1)u] & \chi_2[\bar{\eta}_y a-(\gamma-1)v] & \chi_2(\gamma-1)\end{pmatrix} \tag{B6}$$

with $\bar{\eta}_x=\eta_x/k_2$, $\bar{\eta}_y=\eta_y/k_2$ and $\bar{\theta}_2=\theta_2/k_2$.

Appendix C: Slip-wall boundary condition

Here we introduce for the two-dimensional Euler equations the slip-wall boundary condition imposed through SATs proposed in Ref. [24]. Consider a Cartesian grid and suppose that we are considering a point $(1, j)$ situated on the wall boundary. Taking Eq. (11) as an example, *i.e.*,

$$\frac{\mathrm{d}}{\mathrm{d}t}U_{1,j}+(F_x)_{1,j}+(G_y)_{1,j}=-\frac{\tau_1}{h_x}\bar{\boldsymbol{A}}^{+}_{1,j}(U_{1,j}-C^{w}_{j}) \qquad \text{(C1)}$$

with $\bar{\boldsymbol{A}}^{+}_{1,j}=\bar{\boldsymbol{K}}_{1,j}\bar{\boldsymbol{\Lambda}}^{+}_{1,j}\bar{\boldsymbol{K}}^{-1}_{1,j}$. What we need to do is just to set the normal velocity (here on the Cartesian grid refers to u) as zero, *i.e.*, $\bar{\boldsymbol{\Lambda}}^{+}_{1,j}=\mathrm{diag}[0,0,0,a]$. Then the wall boundary value C^{w}_{j} is constructed by subtracting the normal velocity from $U_{1,j}$ (see Ref. [24] for more information). The implementation of the slip-wall condition for the case with curvilinear grids is similar. Hence the description is omitted here.

References

[1] Deng X G, Jiang Y, Mao M L, Liu H Y, Li S and Tu G H 2015 *Comput. Fluids* **116** 29
[2] Jiang Y, Mao M L, Deng X G and Liu H Y 2014 *Comput. Fluids* **104** 73
[3] Deng X G and Chen Y M 2018 *J. Comput. Phys.* **372** 80
[4] Wamg Z D, Yang J F, Wei Y K, *et al.* 2013 *Chin. Phys. Lett.* **30** 094703
[5] Sun D K, Jiang D, Xiang N, *et al.* 2013 *Chin. Phys. Lett.* **30** 074702
[6] Qin J X, Chen Y M and Deng X G 2019 *Appl. Math. Mech-Engl.* **40** 823
[7] Carpenter M H, Gottlieb D and Abarbanel S 1994 *J. Comput. Phys.* **111** 220
[8] Hesthaven J S 1998 *SIAM J. Sci. Comput.* **20** 62
[9] Kopriva D A and Kolias J H 1996 *J. Comput. Phys.* **125** 224
[10] Kozdon J E and Wilcox L C 2015 *SIAM J. Sci. Comput.* **38** A923
[11] Sun Y Z, Wang Z J and Liu Y 2007 *Commun. Comput. Phys.* **2** 310
[12] Carpenter M H, Nordström J and Gottlieb D 1999 *J. Comput. Phys.* **148** 341
[13] Carpenter M H, Nordström J and Gottlieb D 2010 *J. Sci. Comput.* **45** 118
[14] Lundquist T, Malan A and Nordström J 2018 *J. Comput. Phys.* **362** 49
[15] Gong J and Nordström J 2011 *J. Comput. Appl. Math.* **236** 602
[16] Virta K and Mattsson K 2014 *J. Sci. Comput.* **61** 90
[17] Wang S Y, Virta K and Kreiss G 2016 *J. Sci. Comput.* **68** 1002
[18] Toro E 1999 *Riemann Solvers and Numerical Method for Fluid Dynamics — A Practical Introduction* (Berlin: Springer) pp. 2–5
[19] Fisher T C, Carpenter M H, Yamaleev N K and Frankel S H 2011 *J. Comput. Phys.* **230** 3727
[20] Wang, L and Yu, M 2018 *J. Sci. Comput.* **75** 253
[21] Chen Y M and Deng X G 2019 *Comput. Fluids* **185** 13
[22] Casper J, Shu C W and Atkins H 1994 *AIAA Journal* **32** 1970
[23] Dan X, Deng X G, Chen Y M, Dong Y D and Wang G X 2016 *Comput. Fluids* **129** 20
[24] Svärd M and Nordström J 2008 *J. Comput. Phys.* **227** 4805