Journal of Computational Physics 365 (2018) 206–225

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

An exponential time-integrator scheme for steady and unsteady inviscid flows

Shu-Jie Li [a], Li-Shi Luo [a,b,*], Z.J. Wang [c], Lili Ju [d]

[a] Beijing Computational Science Research Center, Beijing 100193, China
[b] Department of Mathematics and Statistics, Old Dominion University, Norfolk, VA 23529, USA
[c] Department of Aerospace Engineering, University of Kansas, Lawrence, KS 66045, USA
[d] Department of Mathematics, University of South Carolina, Columbia, SC 29208, USA

ARTICLE INFO

Article history:
Received 23 April 2017
Received in revised form 6 March 2018
Accepted 10 March 2018
Available online 20 March 2018

Keywords:
Exponential time integration
Predictor–corrector method
Large time step
Discontinuous Galerkin
Unstructured meshes
Compressible flow

ABSTRACT

An exponential time-integrator scheme of second-order accuracy based on the predictor–corrector methodology, denoted PCEXP, is developed to solve multi-dimensional nonlinear partial differential equations pertaining to fluid dynamics. The effective and efficient implementation of PCEXP is realized by means of the Krylov method. The linear stability and truncation error are analyzed through a one-dimensional model equation. The proposed PCEXP scheme is applied to the Euler equations discretized with a discontinuous Galerkin method in both two and three dimensions. The effectiveness and efficiency of the PCEXP scheme are demonstrated for both steady and unsteady inviscid flows. The accuracy and efficiency of the PCEXP scheme are verified and validated through comparisons with the explicit third-order total variation diminishing Runge–Kutta scheme (TVDRK3), the implicit backward Euler (BE) and the implicit second-order backward difference formula (BDF2). For unsteady flows, the PCEXP scheme generates a temporal error much smaller than the BDF2 scheme does, while maintaining the expected acceleration at the same time. Moreover, the PCEXP scheme is also shown to achieve the computational efficiency comparable to the implicit schemes for steady flows.

1. Introduction

Significant progress has been made recently in the development of high-order spatial discretization methods in computational fluid dynamics (CFD), such as the discontinuous Galerkin (DG) [1–8], multi-moment constrained finite-volume (MCV) [9], flux reconstruction (FR) or correction procedure *via* reconstruction (CPR) method [10–12], and others [13–15]. These high-order techniques have exhibited a great potential as effective numerical solution methods amenable for efficient implementation on massively parallel high-performance computers. For complex geometries, an efficient solution, however, also depends on the availability of a fast time advancement solver. In contrast to a relative ubiquity of efficient techniques for spatial discretizations, efficient time-marching approaches for both steady and unsteady flows seem to be limited. Efficient time-integration approaches are thus the focus of the present work.

For unsteady flows, explicit methods, such as Runge–Kutta (RK) approaches are prevalent for their simplicity. However, with highly clustered nonuniform meshes, the Courant–Friedrichs–Lewy (CFL) condition can severely limit the local time-

* Corresponding author at: Beijing Computational Science Research Center, Beijing 100193, China and Department of Mathematics and Statistics, Old Dominion University, Norfolk, VA 23529, USA.
E-mail addresses: shujie@csrc.ac.cn (S.-J. Li), luols1989@gmail.com (L.-S. Luo), zjw@ku.edu (Z.J. Wang), ju@math.sc.edu (L. Ju).

https://doi.org/10.1016/j.jcp.2018.03.020
0021-9991/

step size. The restriction due to the CFL condition is particularly acute for direct numerical simulation (DNS) and large-eddy simulation (LES) of turbulent flows, which usually require very fine grids of high aspect ratios in near-wall regions. Thus, the restriction due to the CFL condition becomes a critical bottleneck in computational efficiency for explicit time-marching schemes.

To enhance the computational efficiency of explicit time-marching schemes, it is desirable to relax or to remove the limitation of the CFL condition. To this end, a class of schemes based on the exponential time integration shows a great potential [16–29]. In contrast to usual explicit time-marching schemes, these schemes allow much larger time-step sizes while maintaining excellent numerical stability.

In explicit time-marching methods, information cannot propagate beyond the localities constrained by the CFL condition in each time step. In exponential time-marching methods, however, information is propagated to the entire computational domain instantaneously through the global Jacobian, similar to implicit methods, thus significantly alleviating the restriction on time-step size imposed by the CFL condition, if not eliminating it altogether. As mentioned previously, a variety of schemes based on the exponential integration have been developed already (cf., *e.g.*, [16–29]). While the basic idea of exponential integration has been adopted in the aforementioned methods, the existing algorithms differ from each other in some aspects. There are two types of exponential schemes depending on the treatment of the nonlinear term, *i.e.*, explicit and implicit. The classic ETD scheme is a typical explicit scheme (cf., *e.g.*, [16]), while the implicit integrator factor method is an implicit one (cf., *e.g.*, [21–23]), which can usually alleviate the stiffness due to the nonlinear term but requires nonlinear iterations in each time step.

While most of the exponential schemes are applied to specialized equations [16–27] with either scalar exponentials or constant matrix exponentials, such as the applications to semilinear parabolic equations [28,29], and relatively few are applied to practical CFD problems (cf., *e.g.*, [30–32]) with time-dependent full matrix exponential computations. There are some key issues, such as the computational efficiency for steady problems and the temporal accuracy for unsteady problems, have yet to be fully investigated. The overarching goal of the present work is to develop an efficient and time-accurate exponential scheme to solve multi-dimensional fluid dynamic equations. Specifically, we develop a second-order exponential time-integrator scheme to solve the Euler equations for steady and unsteady problems in both two and three dimensions, and assess its accuracy and computational efficiency by comparing with several well-known explicit and implicit approaches.

The remainder of this paper is organized as follows. Section 2 discusses the construction of a second-order exponential scheme based on the predictor–corrector methodology, denoted as PCEXP, and its efficient implementation through the Krylov method. Section 3 describes a linear stability and error analysis of PCEXP for a simple model equation in one dimension. Section 4 presents the application of PCEXP to the Euler equations discretized with a high-order DG method in space. Section 5 presents the numerical results of this work including three inviscid flow problems: (a) the transportation of an isentropic vortex in 2D with a constant velocity; (b) subsonic flow over a NACA0012 airfoil with a Mach number $\mathrm{Ma} = 0.63$; and (c) subsonic flow over a sphere in 3D with $\mathrm{Ma} = 0.3$. The numerical results obtained with PCEXP are compared with third-order Total Variation Diminishing Runge–Kutta scheme (TVDRK3), implicit backward Euler (BE), and second-order backward difference formula (BDF2). Finally, Section 6 summarizes and concludes this work. Appendix A provides the details of the Jacobian matrices.

2. Exponential time-integrator schemes

In this section, we first develop a predictor–corrector based the second-order exponential time-integrator scheme, and then discuss the efficient implementation through the Krylov method. We also carry out a linear stability analysis of the proposed scheme applied to a model equation in 1D to demonstrate its feasibility of time marching with large time steps.

2.1. Predictor–corrector exponential time-integrator scheme (PCEXP)

We start with the following semi-discrete system of autonomous ordinary differential equations which may be obtained from a spatial discretization:

$$\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} = \mathbf{R}(\mathbf{u}), \tag{1}$$

where $\mathbf{u} = \mathbf{u}(t) \in \mathbb{R}^K$ denotes the vector of the solution variables and $\mathbf{R}(\mathbf{u}) \in \mathbb{R}^K$ the right-hand-side term which may be the spatially discretized residual terms of the discontinuous Galerkin method used in this work. The dimension K is the degrees of freedom which can be very large for 3D problems. Without loss of generality, we consider $\mathbf{u}(t)$ in the interval of one time step, *i.e.*, $t \in [t_n, t_{n+1}]$.

We apply the term splitting method [24] to treat Eq. (1):

$$\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} = \mathbf{J}_n\mathbf{u} + \mathbf{N}(\mathbf{u}), \tag{2}$$

where the subscript n indicates the value evaluated at $t = t_n$, $\mathbf{J}_n$ denotes the Jacobian matrix $\mathbf{J}_n := \partial\mathbf{R}(\mathbf{u})/\partial\mathbf{u}|_{t=t_n} := \partial\mathbf{R}(\mathbf{u}_n)/\partial\mathbf{u}$, $\mathbf{u}_n := \mathbf{u}(t_n)$, and $\mathbf{N}(\mathbf{u}) := \mathbf{R}(\mathbf{u}) - \mathbf{J}_n\mathbf{u}$ denotes the remainder, which in general is nonlinear. Equation (2) admits the following formal solution:

$$\mathbf{u}_{n+1} = \exp(\Delta t \mathbf{J}_n)\mathbf{u}_n + \int_0^{\Delta t} \exp\left((\Delta t - \tau)\mathbf{J}_n\right)\mathbf{N}(\mathbf{u}(t_n + \tau))\,\mathrm{d}\tau, \tag{3}$$

where $\Delta t := t_{n+1} - t_n$ and

$$\exp(-t\mathbf{J}_n) := \sum_{m=0}^{\infty} \frac{(-t\mathbf{J}_n)^m}{m!} \tag{4}$$

is the integrating factor. The formal solution (3) is the starting point to derive the proposed exponential scheme in which the stiff part is computed analytically whereas the nonlinear term is approximated numerically.

By substituting the nonlinear term $\mathbf{N}(t_n + \tau)$ with its Taylor expansion at t_n

$$\mathbf{N}(\mathbf{u}(t_n + \tau)) = \sum_{k=1}^{\infty} \frac{\tau^{k-1}}{(k-1)!} \frac{\partial^{k-1}\mathbf{N}(\mathbf{u}_n)}{\partial \tau^{k-1}}, \tag{5}$$

the solution (3) becomes

$$\begin{aligned}\mathbf{u}_{n+1} &= \exp(\Delta t\mathbf{J}_n)\,\mathbf{u}_n + \exp(\Delta t\mathbf{J}_n) \sum_{k=1}^{\infty} \frac{1}{(k-1)!} \left[\int_0^{\Delta t} \exp(-\tau\mathbf{J}_n)\,\tau^{k-1}\,\mathrm{d}\tau\right] \frac{\partial^{k-1}\mathbf{N}(\mathbf{u}_n)}{\partial\tau^{k-1}} \\ &= \exp(\Delta t\mathbf{J}_n)\,\mathbf{u}_n + \sum_{k=1}^{\infty} \Delta t^k\,\boldsymbol{\Phi}_k(\Delta t\mathbf{J}_n) \frac{\partial^{k-1}\mathbf{N}(\mathbf{u}_n)}{\partial\tau^{k-1}},\end{aligned} \tag{6}$$

where the tensorial function $\boldsymbol{\Phi}_k(\Delta t\mathbf{J}_n)$ is defined as the following:

$$\boldsymbol{\Phi}_k(\Delta t\mathbf{J}_n) := \frac{\exp(\Delta t\mathbf{J}_n)}{\Delta t^k (k-1)!} \int_0^{\Delta t} \exp(-\tau\mathbf{J}_n)\tau^{k-1}\,\mathrm{d}\tau, \quad k \geq 1, \tag{7}$$

and it satisfies the following recursion relationship:

$$\boldsymbol{\Phi}_{k+1}(\Delta t\mathbf{J}) = \frac{\mathbf{J}^{-1}}{\Delta t\, k!}\left[k!\,\boldsymbol{\Phi}_k(\Delta t\mathbf{J}) - \mathbf{I}\right], \quad k \geq 1, \tag{8a}$$

$$\boldsymbol{\Phi}_1(\Delta t\mathbf{J}) := \frac{\mathbf{J}^{-1}}{\Delta t}\left[\exp(\Delta t\mathbf{J}) - \mathbf{I}\right], \tag{8b}$$

where $\mathbf{I}$ denotes the $K \times K$ identity matrix. Thus, an approximation of the integral in (3) by a truncated Taylor expansion of the nonlinear term $\mathbf{N}$ leads to an exponential scheme consisting of a linear combination of products linear in $\boldsymbol{\Phi}_k$. Specifically, with $k = 1$, the nonlinear term $\mathbf{N}$ is approximated by a constant, *i.e.*, its left-end value $\mathbf{N}_n$ on the interval $[t_n, t_{n+1}]$, hence,

$$\mathbf{N}(\mathbf{u}(t_n + \tau)) \approx \mathbf{N}_n = \mathbf{R}_n - \mathbf{J}_n\mathbf{u}_n, \tag{9}$$

leading to a simple exponential scheme

$$\mathbf{u}_{n+1} = \exp(\Delta t\mathbf{J}_n)\,\mathbf{u}_n + \Delta t\,\boldsymbol{\Phi}_1(\Delta t\mathbf{J}_n)\,\mathbf{N}_n = \mathbf{u}_n + \Delta t\,\boldsymbol{\Phi}_1(\Delta t\mathbf{J}_n)\,\mathbf{R}_n, \tag{10}$$

which is the first-order exponential-time differencing scheme ETD1 [16], also referred as exponential Rosenbrock–Euler method [24].

With $k = 2$, a first-order finite-difference approximation to the derivative of $\mathbf{N}$, *i.e.*,

$$\partial_\tau \mathbf{N}_n \approx \frac{(\mathbf{N}_n - \mathbf{N}_{n-1})}{\Delta t},$$

leads to the second-order scheme ETD2 [28]:

$$\mathbf{u}_{n+1} = \underbrace{\exp(\Delta t\mathbf{J}_n)\,\mathbf{u}_n + \Delta t\,\boldsymbol{\Phi}_1(\Delta t\mathbf{J}_n)\,\mathbf{N}_n}_{\text{ETD1 defined by (10)}} + \Delta t\,\boldsymbol{\Phi}_2(\Delta t\mathbf{J}_n)\,(\mathbf{N}_n - \mathbf{N}_{n-1}), \tag{11a}$$

$$\boldsymbol{\Phi}_2(\Delta t\mathbf{J}) := \frac{\mathbf{J}^{-2}}{\Delta t^2}\left[\exp(\Delta t\mathbf{J}) - \Delta t\mathbf{J} - \mathbf{I}\right]. \tag{11b}$$

The ETD2 scheme is in fact the ETD1 scheme with an additional term $\mathbf{\Phi}_2(\Delta t\mathbf{J}_n)(\mathbf{N}_n - \mathbf{N}_{n-1})$. It should be stressed that the calculation of $\mathbf{\Phi}_2$ is computationally more demanding than that of $\mathbf{\Phi}_1$ for a system with large degrees of freedom K. Hence, the ETD2 scheme may not be practical for large systems.

The Runge–Kutta or multi-step approximations for the nonlinear term $\mathbf{N}(\mathbf{u})$ can also be used to construct high-order schemes (cf., *e.g.*, [16,26,29]). However, the objective of this work is to construct an effective and efficient second-order ETD scheme which only requires $\mathbf{\Phi}_1$. To this end, we design a scheme based on the idea of the predictor–corrector methodology consisting of two stages. First, the solution is advanced with the first-order ETD scheme (10) to obtain a predicted solution $\mathbf{u}_*$. Next, the solution $\mathbf{u}_{n+1}$ is corrected by replacing the nonlinear term $\mathbf{N}_n := \mathbf{N}(\mathbf{u}_n)$ by the algebraic average of itself and its predicted solution $\mathbf{N}_* := \mathbf{N}(\mathbf{u}_*)$, which is a standard second-order Gaussian quadrature or midpoint approximation. This simple procedure enhances the accuracy of the scheme from first order to second order. The two-stage scheme can be summarized below:

$$\mathbf{u}_* = \mathbf{u}_n + \Delta t\, \mathbf{\Phi}_1(\Delta t\mathbf{J}_n)\mathbf{R}_n, \tag{12a}$$

$$\mathbf{u}_{n+1} = \mathbf{u}_* + \frac{1}{2}\Delta t\, \mathbf{\Phi}_1(\Delta t\mathbf{J}_n)\,(\mathbf{N}_* - \mathbf{N}_n)\,. \tag{12b}$$

The above two-stage scheme is designated as the predictor–corrector exponential (PCEXP) scheme. The first stage of PCEXP is designated as EXP1, which is only used for steady problems. Note that the PCEXP scheme is in fact a one-step scheme, *i.e.*, it only requires the solution $\mathbf{u}$ at the current time $t = t_n$.

2.2. Realization of PCEXP with the Krylov method

The exponential time-integrator schemes require evaluations of matrix–vector products, and in particular, the product of the exponential functions of the Jacobian and a vector, *e.g.*, $\mathbf{\Phi}_1(\Delta t\mathbf{J}_n)\mathbf{R}$ in (12a) or $\mathbf{\Phi}_1(\Delta t\mathbf{J}_n)\mathbf{N}$ in (12b). If the inverse of the Jacobian $\mathbf{J}$ exists, then it is possible to use $\mathbf{J}^{-1}$ to compute $\mathbf{\Phi}_1(\Delta t\mathbf{J})$ defined by (8b). However, $\mathbf{J}$ may be singular, *e.g.*, in the presence of periodic boundary conditions, thus $\mathbf{J}^{-1}$ may not be computable. In addition, for a problem with a very large number of degrees of freedom, direct inversion of $\mathbf{J}$ can be prohibitively expensive to compute. These impediments could be one reason why the exponential schemes have yet to gain much traction in the realm of realistic CFD applications.

The matrix–vector products in (12b) can be approximated efficiently using the Krylov method [33,34], which can also treat a singular $\mathbf{J}$. The basic idea of the Krylov method is to approximate the product of $\exp(\Delta t\mathbf{J})$ and a vector, such as $\mathbf{N}$ in (12b), by projecting it onto a small Krylov subspace, resulting in a much smaller matrix thus cheaper in computational effort. The algorithm will be discussed in detail next.

With the Taylor expansion of $\exp(\Delta t\mathbf{J})$, the product $\mathbf{\Phi}_1\mathbf{N}$ can be written as:

$$\mathbf{J}^{-1}\frac{\exp(\Delta t\mathbf{J}) - \mathbf{I}}{\Delta t}\mathbf{N} = \sum_{k=0}^{\infty}\frac{(\Delta t\mathbf{J})^k}{(k+1)!}\mathbf{N} = \left(\mathbf{I} + \frac{(\Delta t\mathbf{J})}{2!} + \frac{(\Delta t\mathbf{J})^2}{3!} + \cdots\right)\mathbf{N}. \tag{13}$$

It can be approximated by the following function projection onto the Krylov subspace of dimension m:

$$\mathbb{K}_m(\mathbf{J}, \mathbf{N}) = \text{span}\{\mathbf{N},\ \mathbf{J}\mathbf{N},\ \mathbf{J}^2\mathbf{N},\ \ldots,\ \mathbf{J}^{m-1}\mathbf{N}\}. \tag{14}$$

The orthonormal basis $\mathbb{V}_m = \{\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_m\}$ of the Krylov subspace $\mathbb{K}_m$, with $\mathbf{v}_i \cdot \mathbf{v}_j = \delta_{ij}$, can be constructed with the well known Arnoldi's algorithm (cf., *e.g.*, [35]).

The orthogonal basis matrix $\mathbf{V}_m := (\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_m) \in \mathbb{R}^{K\times m}$ satisfies the so-called Arnoldi decomposition [34]:

$$\mathbf{J}\mathbf{V}_m = \mathbf{V}_{m+1}\widetilde{\mathbf{H}}_m, \tag{15}$$

where $\mathbf{V}_{m+1} := (\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_m, \mathbf{v}_{m+1}) = (\mathbf{V}_m, \mathbf{v}_{m+1}) \in \mathbb{R}^{K\times(m+1)}$ and $\widetilde{\mathbf{H}}_m$ is the following $(m+1)\times m$ upper-Hessenberg matrix:

$$\widetilde{\mathbf{H}}_m = \begin{bmatrix} h_{1,1} & h_{1,2} & h_{1,3} & h_{1,4} & \cdots & h_{1,m} \\ h_{2,1} & h_{2,2} & h_{2,3} & h_{2,4} & \cdots & h_{2,m} \\ 0 & h_{3,2} & h_{3,3} & h_{3,4} & \cdots & h_{3,m} \\ & 0 & h_{4,3} & \ddots & \ddots & \vdots \\ \vdots & & 0 & \ddots & h_{m-1,m-1} & h_{m-1,m} \\ & & & \ddots & h_{m,m-1} & h_{m,m} \\ 0 & & \cdots & & 0 & h_{m+1,m} \end{bmatrix}. \tag{16}$$

The upper-Hessenberg matrix $\widetilde{\mathbf{H}}_m$ can be written as the following:

$$\widetilde{\mathbf{H}}_m = \begin{bmatrix} \mathbf{H}_m \\ h_{m+1,m}\mathbf{e}_m^{\mathsf{T}} \end{bmatrix}, \tag{17}$$

where $\mathbf{H}_m$ is the matrix composed of the first m rows of $\widetilde{\mathbf{H}}_m$ and $\mathbf{e}_m := (0, \cdots, 0, 1)^{\mathbf{T}} \in \mathbb{R}^m$ is the m-th canonical basis vector in $\mathbb{R}^m$, then Eq. (15) becomes

$$\mathbf{J}\mathbf{V}_m = \mathbf{V}_m\mathbf{H}_m + h_{m+1,m}\mathbf{v}_{m+1}\mathbf{e}_m^{\mathbf{T}}. \tag{18}$$

Because $\mathbf{V}_m^{\mathbf{T}}\mathbf{V}_m = \mathbf{I}$, therefore

$$\mathbf{H}_m = \mathbf{V}_m^{\mathbf{T}}\mathbf{J}\mathbf{V}_m, \tag{19}$$

that is, $\mathbf{H}_m$ is the projection of the linear transformation of $\mathbf{J}$ onto the subspace $\mathbb{K}_m$ with the basis $\mathbb{V}_m$. Since $\mathbf{V}_m\mathbf{V}_m^{\mathbf{T}} \neq \mathbf{I}$, Eq. (19) leads to the following approximation:

$$\mathbf{J} \approx \mathbf{V}_m\mathbf{V}_m^{\mathbf{T}}\mathbf{J}\mathbf{V}_m\mathbf{V}_m^{\mathbf{T}} = \mathbf{V}_m\mathbf{H}_m\mathbf{V}_m^{\mathbf{T}}, \tag{20}$$

and $\exp(\mathbf{J})$ can be approximated by $\exp(\mathbf{V}_m\mathbf{H}_m\mathbf{V}_m^{\mathbf{T}})$ as the following:

$$\exp(\mathbf{J})\mathbf{N} \approx \exp(\mathbf{V}_m\mathbf{H}_m\mathbf{V}_m^{\mathbf{T}})\mathbf{N} = \mathbf{V}_m\exp(\mathbf{H}_m)\mathbf{V}_m^{\mathbf{T}}\mathbf{N}. \tag{21}$$

The first column vector of $\mathbf{V}_m$ is $\mathbf{v}_1 = \mathbf{N}/\|\mathbf{N}\|_2$ and $\mathbf{V}_m^{\mathbf{T}}\mathbf{N} = \|\mathbf{N}\|_2\,\mathbf{e}_1$, thus (21) becomes:

$$\exp(\mathbf{J})\mathbf{N} \approx \|\mathbf{N}\|_2\mathbf{V}_m\exp(\mathbf{H}_m)\,\mathbf{e}_1. \tag{22}$$

Consequently $\mathbf{\Phi}_1$ can be approximated by:

$$\mathbf{\Phi}_1(\Delta t\mathbf{J})\mathbf{N} = \frac{1}{\Delta t}\int_0^{\Delta t}\exp((\Delta t-\tau)\mathbf{J})\mathbf{N}\,\mathrm{d}\tau \approx \frac{1}{\Delta t}\int_0^{\Delta t}\|\mathbf{N}\|_2\mathbf{V}_m\exp\left((\Delta t-\tau)\mathbf{H}_m\right)\,\mathbf{e}_1\,\mathrm{d}\tau. \tag{23}$$

In general, the dimension of the Krylov subspace, m, is chosen to be much smaller than the dimension of $\mathbf{J}$, K, thus $\mathbf{H}_m \in \mathbb{R}^{m\times m}$ can be inverted easily, so $\mathbf{\Phi}_1$ can be easily computed as the following:

$$\begin{aligned}\mathbf{\Phi}_1(\Delta t\mathbf{J})\mathbf{N} &\approx \frac{1}{\Delta t}\|\mathbf{N}\|_2\mathbf{V}_m\int_0^{\Delta t}\exp\left((\Delta t-\tau)\mathbf{H}_m\right)\,\mathbf{e}_1\,\mathrm{d}\tau\\ &= \frac{1}{\Delta t}\|\mathbf{N}\|_2\mathbf{V}_m\mathbf{H}_m^{-1}\left[\exp(\Delta t\mathbf{H}_m)-\mathbf{I}\right]\mathbf{e}_1,\end{aligned} \tag{24}$$

where the matrix-exponential $\exp(\Delta t\mathbf{H}_m)$ can be computed efficiently by the Chebyshev rational approximation (cf., *e.g.*, [34,36]) due to the small size of $\mathbf{H}_m$.

3. Linear stability analysis and local truncation error

In this section, we carry out a linear stability analysis of the proposed PCEXP scheme by considering a scalar equation for which analytic results can be obtained. This example is instructive because the growth rate in the scalar equation, $\exp(\Delta t\,J)$ with a constant J, is the degenerated case of $\exp(\Delta t\mathbf{J})$. The stability of the PCEXP scheme is compared with the TVDRK3 scheme. In addition, we will also analyze the local truncation error of the PCEXP scheme applied to the scalar equation.

3.1. Linear stability analysis

We shall analyze the stability of the following scalar equation with a constant J:

$$u' = Ju + N(u). \tag{25}$$

Linearization of equation (25) about a fixed point u_0, such that $Ju_0 + N(u_0) = 0$, leads to the following simple linear equation:

$$u' = J\,u + \lambda\,u := \zeta\,u, \tag{26}$$

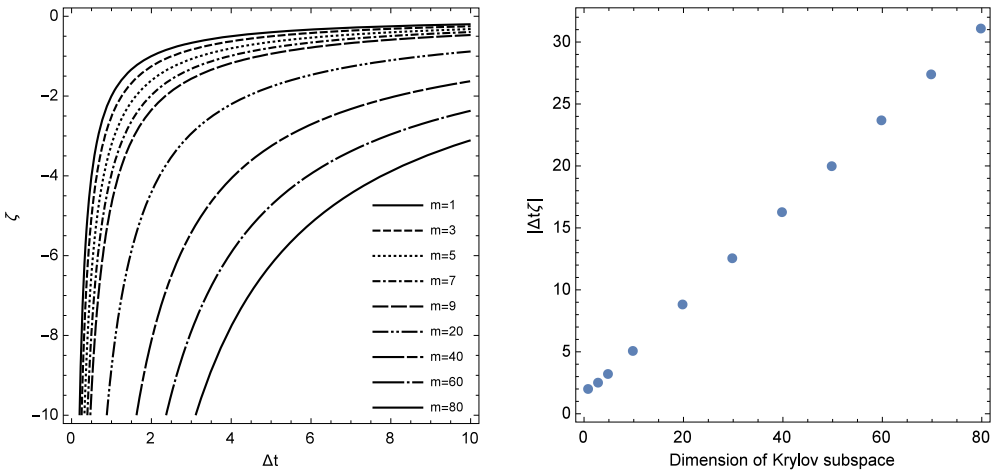
where u is now the perturbation to u_0, $\lambda = N'(u_0)$, and $\zeta := J + \lambda$. Obviously, the fixed point u_0 is stable if and only if

$$\mathrm{Re}(\zeta) \equiv \mathrm{Re}(J+\lambda) \leq 0. \tag{27}$$

To analyze the dependence of the stability region on the finite-term Krylov basis approximation, we apply the PCEXP scheme (12b) to the linearized model equation (26) and obtain the following very simple solution:

$$u_{n+1} = \exp(\Delta t\zeta)\,u_n := G\,u_n, \quad G := \exp(\Delta t\zeta). \tag{28}$$

Note that the solution of the linear equation produced by the PCEXP scheme is the exact solution of the same linear equation in a standard exponential form. This capability of producing the exact solution of a linear equation is an important



(a) m-dependence of stability curves (b) m-dependence of $|\Delta t\zeta|$

Fig. 1. The m-dependence of the stability boundary in the parameter space $(\Delta t,\ \zeta)$ (left) and the stability constant $|\Delta t\,\zeta|$ (right). An m-dependent stable region is bounded by the both axes of Δt and ζ and an m-dependent curve in the fourth quadrant on $(\Delta t,\ \zeta)$.

Table 1
Time step with various number of Krylov basis.

m	Δt	$\Delta t/\Delta t_1$	m	Δt	$\Delta t/\Delta t_1$
1	2.00000/|λ|	1.00	40	16.2705/|λ|	8.14
5	3.21705/|λ|	1.61	50	19.9819/|λ|	9.99
10	5.06952/|λ|	2.53	60	23.6883/|λ|	11.84
20	8.82143/|λ|	4.41	70	27.3910/|λ|	13.70
30	12.55170/|λ|	6.28	80	31.0908/|λ|	15.54

feature of exponential schemes. The function G can be approximated by its Taylor expansion up to m-th order:

$$G_m = 1 + (\Delta t\zeta) + \frac{(\Delta t\zeta)^2}{2!} + \frac{(\Delta t\zeta)^3}{3!} + \cdots + \frac{(\Delta t\zeta)^m}{m!}. \tag{29}$$

The polynomial G_m approximates the growth rate G, and the stability criterion (27) requires that $|G_m| \le 1$.

We compute the dependence of the stability region determined by $|G_m| \le 1$ in the parameter space $(\Delta t,\ \zeta)$ on the order of the polynomial G_m, m. First, for real ζ, we compute the boundary of the stability region with $1 \le m \le 80$. The m-dependence of the stability boundary in the parameter space $(\Delta t,\ \zeta)$ is shown in Fig. 1(a); the stability region is bounded by both Δt and ζ axes and the m-dependent boundary. Clearly, the stability region expands as m increases. We also compute the m-dependence of the maximum stable value of $|\Delta t\zeta|$, as shown in Fig. 1(b). It can be seen that the maximum stable value of $|\Delta t\zeta|$ increases with m linearly, for the linear problem considered. The result of Fig. 1(b) is also tabulated in Table 1, which also gives the m-dependent time-step sizes normalized by Δt_1 corresponding to $m = 1$.

For complex ζ, the boundary of the stability region is defined in the complex plane of $\Delta t\zeta$. As shown in Fig. 2, the stability region is approximately a semicircle on the left half of the complex plane ζ, and the radius of the semicircle grows linearly as m increases.

The preceding analysis shows the effect of the number of the terms in the Taylor expansion of the propagator, $\exp(\Delta t\zeta)$, on the stability of the exponential scheme. Next, we analyze the stability of the PCEXP scheme (12b) by using the nonlinear model problem (25). Applying the PCEXP scheme (12b) to the nonlinear model problem (25) with $N(u) = \lambda u$ leads to the following growth factor:

$$\begin{aligned} r = \frac{u_{n+1}}{u_n} &= \frac{\lambda\,(\lambda - J) + \lambda\,(\lambda + J)\,e^{2J\Delta t} + 2(J^2 - \lambda^2)\,e^{J\Delta t}}{2J^2} \\ &= \frac{a(a-b) + a(a+b)\,e^{2b} + 2(b^2 - a^2)\,e^b}{2b^2}, \end{aligned} \tag{30}$$

where $a := \Delta t\,\lambda$ and $b := \Delta t\ J$. The Taylor expansion of r in terms of b is

$$r = \left(1 + a + \frac{a^2}{2}\right) + \left(1 + a + \frac{a^2}{2}\right) b + \left(\frac{1}{2} + \frac{2a}{3} + \frac{7a^2}{24}\right) b^2 + O(b^3). \tag{31}$$

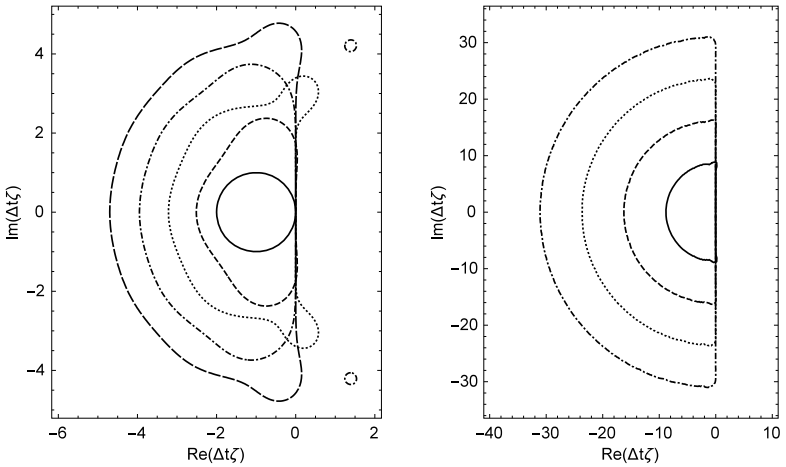


Fig. 2. The m-dependence of the stability region for the scalar equation (26) on the complex $\Delta t\,\zeta$ plane ($\zeta \in \mathbb{C}$). Left: $m = 1, 3, 5, 7, 9$; Right: $m = 20, 40, 60, 80$. The stability region expands monotonically as m increases.

In the limit that $b \to 0$, the growth factor r converges to the leading term of (31)

$$r \to 1 + a + \frac{a^2}{2}, \tag{32}$$

which is identical to the second-order Runge–Kutta scheme.

The above analysis directly shows the following two important features of the PCEXP scheme in the limits of $N \to 0$ and $J \to 0$:

1. The PCEXP scheme reduces to the exact solution of the linear equation in the limit of vanishing nonlinear term $N \to 0$, as shown by (28);
2. The PCEXP scheme converges to the second-order Runge–Kutta scheme in the limit of vanishing linear term $J \to 0$, as shown by (32).

We now consider the dependence of the stability region on the parameter $a := \Delta t\,\lambda$ and $b := \Delta t\,J$. For real a and b, the boundary of the stability region is defined by $|r| = 1$. For the PCEXP scheme, r is given by (30), and $|r| \le 1$ leads to

$$a \le -b, \quad a \ge \frac{2b}{1 - e^b}. \tag{33}$$

For the EXP1 scheme,

$$r = \frac{u_{n+1}}{u_n} = \frac{a(e^b - 1) + be^b}{b}, \tag{34}$$

then the corresponding stability region is bounded by

$$a \le -b, \quad a \ge -b\,\frac{e^b + 1}{e^b - 1} = -b \coth \frac{b}{2}. \tag{35}$$

Similarly, for the TVDRK3 scheme

$$r = \frac{u_{n+1}}{u_n} = 1 + (a + b) + \frac{1}{2}(a + b)^2 + \frac{1}{6}(a + b)^3, \tag{36}$$

so its stability region is a strip bounded by two parallel lines:

$$a \le -b, \quad a \ge -b - 1 - \left(4 + \sqrt{17}\right)^{1/3} + \left(4 + \sqrt{17}\right)^{-1/3}. \tag{37}$$

The stability regions of PCEXP, EXP1, and TVDRK3 are illustrated in Fig. 3.

The stability region of the PCEXP scheme is larger than that of the EXP1 scheme, therefore the PCEXP scheme is not only more accurate but also better in terms of stability. Note that the stability regions of EXP1 and PCEXP schemes are infinitely large fan-shaped areas without a lower bound, while the TVDRK3 scheme in Fig. 3 (right) is a narrow strip. This implies that given a negative J and a fixed λ in the stability regions, both EXP1 and PCEXP allow an infinity large Δt, while TVDRK3 does not. This stability feature distinguishes the exponential schemes from the TVDRK3 scheme.

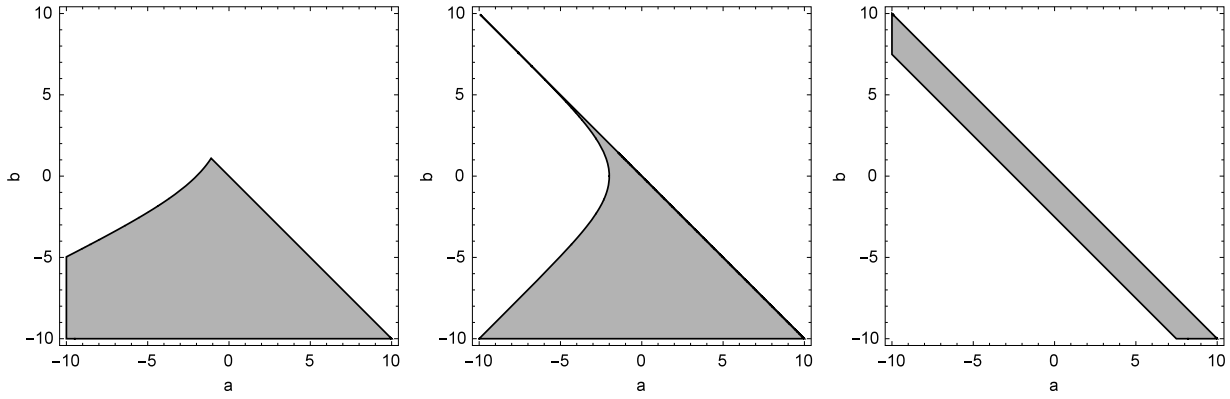


Fig. 3. Stability analysis of exponential and TVDRK3 schemes applied to the linearized scalar equation (26). The stability region (shaded area) within the square $[-10, 10] \times [-10, 10]$ on (a, b) plane. From left to right: PCEXP, EXP1, and TVDRK3.

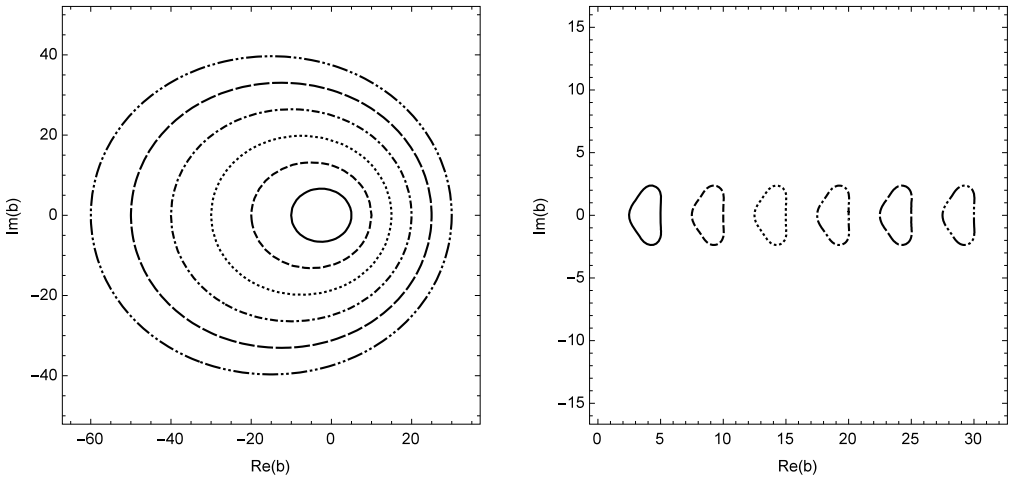


Fig. 4. The stability region of the PCEXP scheme (left) and the TVDRK3 scheme (right) on the complex plan of $b := \Delta t\, J$ with real $a := \Delta t\, \lambda$ as a varying parameter. $a = 5$, 10, 15, 20, 25, and 30, corresponding to enlarging elliptical curves for the PCEXP scheme (left) and the curves moving away positively from the origin along the real axis on the complex b plane for the TVDRK3 scheme (right). The interior regions of the closed curves are the stable regions. Note that the sales of two figures are different.

We can also consider the stability of (30) with a complex $b = \Delta t\, J$ and a real $a = \Delta t\, \lambda$, as shown in Fig. 4. The stability region of the PCEXP scheme increases as $|\lambda|$ increases, while that of the TVDRK3 scheme only shifts a distance along the real axis on the complex b-plane but without changing its area. Thus, the stability region of the TVDRK3 scheme does not expand under the constraint of b. Clearly, the PCEXP scheme is far more superior than the TVDRK3 scheme in terms of stability, as expected.

3.2. Local truncation error

We study the local truncation error of the PCEXP scheme by using the model scalar equation (26). The Taylor expansion of the solution (28) of the linearized equation (26), u_{n+1}, at $t = t_n$ is

$$\begin{aligned} u_{\text{Taylor}} =& u_n + \Delta t\, u_n^{(1)} + \frac{1}{2}\Delta t^2 u_n^{(2)} + \frac{1}{6}\Delta t^3 u_n^{(3)} + O(\Delta t^4) \\ =& \left[1 + \Delta t(J+\lambda) + \frac{1}{2}\Delta t^2 (J+\lambda)^2 + \frac{1}{6}\Delta t^3 (J+\lambda)^3 + \cdots\right] u_n, \end{aligned} \tag{38}$$

where $u_n^{(k)}$ is the k-th order derivative of $u(t)$ evaluated at $t = t_n$, and we have used the fact that for the linearized equation of (26),

$$u_n^{(k)} = J\, u_n^{(k-1)} + \lambda\, u_n^{(k-1)} = (J+\lambda)^k\, u_n. \tag{39}$$

By substitution of the Taylor expansion of the exponential term $e^{\Delta t\zeta}$ in the solution (28) obtained by the PCEXP scheme (12b) yields

$$u_{n+1} = \left[1 + \Delta t(J+\lambda) + \frac{1}{2}\Delta t^2(J+\lambda)^2 + \frac{1}{2}\Delta t^3 J\lambda(J+\lambda) + \cdots\right] u_n. \tag{40}$$

The difference between the Taylor expansion (38) and the approximated solution (40) for u_{n+1} yields the local truncation error:

$$\frac{1}{6}\Delta t^3(J+\lambda)(J^2 - J\lambda + \lambda^2)\, u_n. \tag{41}$$

Similarly, the leading-order local truncation error of the EXP1 scheme is

$$\frac{1}{2}\Delta t^2\lambda(J+\lambda)\, u_n. \tag{42}$$

Obviously, the accuracy of the PCEXP scheme is one-order higher than the ETD1 scheme in the framework of classical truncation error analysis.

4. Spatial discretization

In this section, we apply the PCEXP scheme to the Euler equations discretized with the discontinuous Galerkin (DG) method in space.

4.1. Governing equations

Consider the Euler equations in a rotating frame of reference in d dimensional space:

$$\frac{\partial \mathbf{U}}{\partial t} + \nabla \cdot \mathbf{F} = \mathbf{S}, \tag{43}$$

where $\mathbf{U} \in \mathbb{R}^{d+2}$ stands for the vector of conservative variables, $\mathbf{F} \in \mathbb{R}^{(d+2)\times d}$ the convective flux, and $\mathbf{S} \in \mathbb{R}^{d+2}$ the source term:

$$\mathbf{U} = \begin{pmatrix} \rho \\ \rho \boldsymbol{v} \\ \rho E \end{pmatrix}, \quad \mathbf{F} = \begin{pmatrix} \rho\,(\boldsymbol{v} - \boldsymbol{v}_r)^{\mathsf{T}} \\ \rho\,(\boldsymbol{v} - \boldsymbol{v}_r)\boldsymbol{v}^{\mathsf{T}} + p\,\mathbf{I} \\ \rho\, H\,(\boldsymbol{v} - \boldsymbol{v}_r)^{\mathsf{T}} \end{pmatrix}, \quad \mathbf{S} = \begin{pmatrix} 0 \\ -\rho\,\boldsymbol{\omega} \times \boldsymbol{v} \\ 0 \end{pmatrix}, \tag{44}$$

where $\boldsymbol{v} := (u,\, v,\, w)^{\mathsf{T}}$ is the absolute velocity, $\boldsymbol{\omega} := (\omega_x,\, \omega_y,\, \omega_z)^{\mathsf{T}}$ is the angular velocity of the rotating frame of reference, $\boldsymbol{v}_r := \boldsymbol{\omega} \times \boldsymbol{x}$; ρ, p, and e denote the flow density, pressure, and the specific internal energy; $E = e + \frac{1}{2}||\boldsymbol{v}||^2$ and $H = E + p/\rho$ denote the total energy and total enthalpy, respectively; $\mathbf{I}$ denotes the $d \times d$ unit matrix; and the pressure p is given by the equation of state for a perfect gas:

$$p = \rho\,(\gamma - 1)\, e, \tag{45}$$

where $\gamma = 7/5$ is the ratio of specific heats for perfect gas.

4.2. Discontinuous Galerkin discretization

The computational domain Ω is divided into a set of non-overlapping elements of arbitrary shape. We seek an approximation $\mathbf{U}_h$ in each element $E \in \Omega$ with finite dimensional space of polynomial P^p of order p in the discontinuous finite element space

$$\mathbb{V}_h := \{\psi_i \in L^2(\Omega) : \psi_i|_E \in P^p(\Omega),\ \forall\, E \in \Omega\}. \tag{46}$$

The numerical solution of $\mathbf{U}_h$ can be approximated in the finite element space $\mathbb{V}_h$

$$\mathbf{U}_h(\boldsymbol{x},\, t) = \sum_{j=1}^{n} \mathbf{u}_j(t)\psi_j(\boldsymbol{x}). \tag{47}$$

In the weak formulation, the Euler equations (43) in an element E becomes:

$$\int_E \psi_i \psi_j \mathrm{d}\boldsymbol{x} \frac{\mathrm{d}\mathbf{u}_j}{\mathrm{d}t} = -\int_{\partial E} \psi_i \widetilde{\mathbf{F}} \cdot \hat{\boldsymbol{n}}\, \mathrm{d}\boldsymbol{\sigma} + \int_E (\mathbf{F} \cdot \nabla \psi_i + \psi_i \mathbf{S}) \mathrm{d}\boldsymbol{x} := \mathbf{R}_i, \tag{48}$$

where $\hat{\boldsymbol{n}}$ is the out-normal unit vector of the surface element $\boldsymbol{\sigma}$ with respect to the element E, $\widetilde{\mathbf{F}}$ is the Riemann flux [37], which will be approximated by Roe's flux [38], and the Einstein summation convention is used. For an orthonormal basis $\{\psi_i\}$, the term on the left-hand side of Eq. (48) becomes diagonal, so the system is in the standard ODE form of Eq. (1), thus avoiding solving a linear system as required for a non-orthogonal basis. More importantly, the use of orthogonal basis would yield more accurate solutions, especially for high-order methods with $p \gg 2$.

4.3. Orthogonal basis in the Cartesian coordinates

In this paper, the basis function $\psi_i(\boldsymbol{x})$ is defined on the global Cartesian coordinate $\boldsymbol{x} := (x, y, z)$ rather than on the cell-wise, local reference coordinates. The variable values on the Gaussian quadrature points for computing the surface fluxes can be easily accessed without the Jacobian mapping between the local reference coordinates to the global Cartesian ones [39,40], and it also makes the discontinuous Galerkin method feasible on arbitrary polyhedral grids [41].

A simple choice of the basis function in (47) may be the monomials [42] or Taylor basis [43]. However, in the case of distorted meshes, the non-orthogonality of these basis functions may yield an ill-conditioned mass matrix, resulting in degradation of accuracy and even loss of numerical stability. In this work, to construct an orthonormal basis set $\{\psi_i(\boldsymbol{x})\}$, we start with the normalized monomials $\{\chi_i(\boldsymbol{x})\}$:

$$\{\chi_i(\boldsymbol{x})\} := \left\{ \frac{(x-x_c)^{p_1}(y-y_c)^{p_2}(z-z_c)^{p_3}}{L_x^{p_1} L_y^{p_2} L_z^{p_3}} \,\middle|\, 0 \le p_1,\ p_2,\ p_3;\ p_1+p_2+p_3 \le i-1 \right\}, \tag{49a}$$

$$\{\psi_i(\boldsymbol{x})\} := \left\{ s_i \left[\chi_i(\boldsymbol{x}) + \sum_{j=1}^{i-1} c_{ij} \chi_j(\boldsymbol{x}) \right] \,\middle|\, 1 \le i \le N \right\}, \tag{49b}$$

where $N = (p+1)(p+2)(p+3)/6$ is the total number of basis functions for the p-th order DG approximation in 3D space, and

$$x_c := \frac{1}{|E|}\int_E x\, \mathrm{d}\boldsymbol{x}, \qquad y_c := \frac{1}{|E|}\int_E y\, \mathrm{d}\boldsymbol{x}, \qquad z_c := \frac{1}{|E|}\int_E z\, \mathrm{d}\boldsymbol{x},$$

$$L_x := \frac{1}{2}(x_{\max} - x_{\min}), \qquad L_y := \frac{1}{2}(y_{\max} - y_{\min}), \qquad L_z := \frac{1}{2}(z_{\max} - z_{\min}).$$

With the following definition of the inner product on an element E:

$$\langle f,\, g \rangle_E := \int_E f(\boldsymbol{x})\, g(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x},$$

the coefficients $\{s_i\}$ and $\{c_{ij}\}$ can be computed with the modified Gram-Schmidt (MGS) orthogonalization described in the Algorithm 1.

Algorithm 1 Basis orthonormalization.

```
for i = 1 to N do
  for j = 1 to i − 1 do
    w_ij = ⟨ψ_i, χ_j⟩
    ψ_i = ψ_i − w_ij χ_j
    w_ii = √⟨ψ_i, ψ_i⟩
    ψ_i = ψ_i / w_ii
  end for
end for
```

4.4. Exact Jacobian matrix for the exponential schemes

The convergence rate and stability of the PCEXP scheme rely on the accuracy to approximate the Jacobian matrix $\mathbf{J}$, which is directly determined by the local truncation error of (41). In the PCEXP scheme, the broadcasting of global information is achieved through the exact Jacobian matrix which accurately includes the information of both the interior and the boundary of the elements. The details of computing the exact Jacobian is discussed next.

The diagonal Jacobian can be obtained by taking the derivative of (48) with respect to the $\mathbf{u}_j$ of the host cell with the label "L":

$$\frac{\partial \mathbf{R}_i}{\partial \mathbf{u}_j^{\mathrm{L}}} = -\int_{\partial E} \psi_i \frac{\partial \widetilde{\mathbf{F}}(\mathbf{U}_{\mathrm{L}}, \mathbf{U}_{\mathrm{R}})}{\partial \mathbf{U}_{\mathrm{L}}} \frac{\partial \mathbf{U}_{\mathrm{L}}}{\partial \mathbf{u}_j^{\mathrm{L}}} \,\mathrm{d}\boldsymbol{\sigma} + \int_E \left(\nabla \psi_i \frac{\partial \mathbf{F}}{\partial \mathbf{U}} \frac{\partial \mathbf{U}}{\partial \mathbf{u}_j} + \psi_i \frac{\partial \mathbf{S}}{\partial \mathbf{U}} \frac{\partial \mathbf{U}}{\partial \mathbf{u}_j} \right) \mathrm{d}\boldsymbol{x}$$
$$= -\int_{\partial E} \psi_i^{\mathrm{L}} \psi_j^{\mathrm{L}} \frac{\partial \widetilde{\mathbf{F}}(\mathbf{U}_{\mathrm{L}}, \mathbf{U}_{\mathrm{R}})}{\partial \mathbf{U}_{\mathrm{L}}} \,\mathrm{d}\boldsymbol{\sigma} + \int_E \left(\psi_j \nabla \psi_i \frac{\partial \mathbf{F}}{\partial \mathbf{U}} + \psi_i \psi_j \frac{\partial \mathbf{S}}{\partial \mathbf{U}} \right) \mathrm{d}\boldsymbol{x}. \tag{50}$$

Similarly, the off-diagonal Jacobian can be obtained by taking the derivative of (48) with respect to the $\mathbf{u}_j$ of the neighboring cells around the host cell "L", which is marked with the label "R":

$$\frac{\partial \mathbf{R}_i}{\partial \mathbf{u}_j^{\mathrm{R}}} = -\int_{\partial E} \psi_i \frac{\partial \widetilde{\mathbf{F}}(\mathbf{U}_{\mathrm{L}}, \mathbf{U}_{\mathrm{R}})}{\partial \mathbf{U}_{\mathrm{R}}} \frac{\partial \mathbf{U}_{\mathrm{R}}}{\partial \mathbf{u}_j^{\mathrm{R}}} \,\mathrm{d}\boldsymbol{\sigma} = -\int_{\partial E} \psi_i^{\mathrm{L}} \psi_j^{\mathrm{R}} \frac{\partial \widetilde{\mathbf{F}}(\mathbf{U}_{\mathrm{L}}, \mathbf{U}_{\mathrm{R}})}{\partial \mathbf{U}_{\mathrm{R}}} \,\mathrm{d}\boldsymbol{\sigma}. \tag{51}$$

The Riemann flux Jacobian matrices $\partial \widetilde{\mathbf{F}}(\mathbf{U}_{\mathrm{L}}, \mathbf{U}_{\mathrm{R}})/\partial \mathbf{U}_{\mathrm{L}}$, $\partial \widetilde{\mathbf{F}}(\mathbf{U}_{\mathrm{L}}, \mathbf{U}_{\mathrm{R}})/\partial \mathbf{U}_{\mathrm{R}}$ in (50) and (51) are evaluated exactly through the automatic differentiation (AD), others can be derived easily.

The global Jacobian matrix $\mathbf{J}$ is made of the diagonal and off-diagonal matrices above. When $\boldsymbol{\sigma}$ is an interior face, the flux $\widetilde{\mathbf{F}}(\mathbf{U}_{\mathrm{L}}, \mathbf{U}_{\mathrm{R}})$ is calculated with Roe's Riemann solver [38]. When $\boldsymbol{\sigma}$ is a boundary face with a appropriate boundary condition, one has

$$\widetilde{\mathbf{F}}_{\mathrm{bc}} = \widetilde{\mathbf{F}}(\mathbf{U}_{\mathrm{L}}, \mathbf{U}_{\mathrm{ghost}}), \tag{52}$$

where $\mathbf{U}_{\mathrm{ghost}}$ is a function of $\mathbf{U}_{\mathrm{L}}$ corresponding the boundary condition, and $\widetilde{\mathbf{F}}$ is also consistently computed by the same Roe's Riemann solver used on the interior faces. Then, the boundary flux Jacobian matrix can be expressed as

$$\frac{\partial \widetilde{\mathbf{F}}_{\mathrm{bc}}}{\partial \mathbf{U}_{\mathrm{L}}} = \frac{\partial \widetilde{\mathbf{F}}}{\partial \mathbf{U}_{\mathrm{L}}} + \frac{\partial \widetilde{\mathbf{F}}}{\partial \mathbf{U}_{\mathrm{ghost}}} \frac{\partial \mathbf{U}_{\mathrm{ghost}}}{\partial \mathbf{U}_{\mathrm{L}}}. \tag{53}$$

The Jacobian matrix $\nabla \psi_j \partial \mathbf{F}/\partial \mathbf{U}$ in the volume integration and the source-term Jacobian matrix $\partial \mathbf{S}/\partial \mathbf{U}$ are given by (A.1) and (A.3), respectively, in Appendix A. The Jacobians $\partial \widetilde{\mathbf{F}}/\partial \mathbf{U}_{\mathrm{L}}$, $\partial \widetilde{\mathbf{F}}/\partial \mathbf{U}_{\mathrm{ghost}}$ and $\partial \mathbf{U}_{\mathrm{ghost}}/\partial \mathbf{U}_{\mathrm{L}}$ are obtained exactly by the AD.

5. Numerical results

The PCEXP scheme is tested for the time marching of the Euler equations for both steady and unsteady flows. Its accuracy and efficiency are investigated in the unsteady flow case and compared with two widely-used explicit and implicit schemes: the third-order TVD Runge–Kutta (TVDRK3) scheme and the second-order backward difference formula (BDF2). In the steady flow case, the performance of the PCEXP scheme is also investigated by comparing with the implicit backward Euler (BE) and the implicit BDF2 schemes given below:

$$\text{BE:} \qquad \mathbf{u}_{n+1} = \mathbf{u}_n + \Delta t_n \mathbf{R}(\mathbf{u}_{n+1}), \tag{54}$$

$$\text{BDF2:} \qquad \mathbf{u}_{n+1} = \frac{(1+\alpha_n)^2}{1+2\alpha_n}\mathbf{u}_n - \frac{\alpha_n^2}{1+2\alpha_n}\mathbf{u}_{n-1} + \frac{(1+\alpha_n)}{1+2\alpha_n}\Delta t_n \mathbf{R}(\mathbf{u}_{n+1}), \tag{55}$$

where $\alpha_n := \Delta t_n/\Delta t_{n-1}$, so that a variable time-step size is allowed in the BDF2 scheme [44]. The resulting linear systems are solved by an ILU preconditioned GMRES method. In both the exponential and the implicit methods, the dimension of the Krylov basis $m = 30$. The convergence tolerance of the Krylov subspace is set to 1.0×10^{-5}.

In this work, unless otherwise stated, the time-step size is determined as the following:

$$\Delta t = \frac{\mathrm{CFL}\, h_{3\mathrm{D}}}{(2p+1)\,(\|\boldsymbol{v}\| + c)}, \quad h_{3\mathrm{D}} := 2d\frac{|E|}{|\partial E|}, \tag{56}$$

where CFL is the global Courant–Friedrichs–Lewy (CFL) number, p the order of polynomial in DG, $\boldsymbol{v}$ the local velocity at the cell center, c the speed of sound, d the spatial dimension, $|E|$ and $|\partial E|$ are the volume and the boundary surface area of an element E, respectively; and $h_{3\mathrm{D}}$ represents a characteristic size of a 3D cell determined by the ratio of its volume and surface area. The CFL number is a constant for unsteady flows and a variable for steady flows (cf. Eq. (61a) and pertaining discussion). In what follows, time t, length scales, and all other physical quantities computed in this work will be properly nondimensionalized (cf., *e.g.*, [45]), so they are dimensionless unless otherwise stated.

For the quasi-2D problems, we extrude a 2D mesh to a 3D (quasi-2D) mesh by one layer of grids and use $h_{2\mathrm{D}}$ instead of $h_{3\mathrm{D}}$ to eliminate the effect of the z dimension on obtaining the truly 2D time step. Given the cell size Δz in the z direction, $h_{2\mathrm{D}}$ is determined by

$$\frac{2}{h_{2\mathrm{D}}} = \frac{3}{h_{3\mathrm{D}}} - \frac{1}{\Delta z}. \tag{57}$$

5.1. Temporal accuracy test for an unsteady problem

The vortex transportation by a uniform flow of velocity $(U_\infty, 0)$ [14] is computed to test the temporal accuracy of PCEXP. The initial conditions of the flow are

$$\begin{aligned} U_0 &= U_\infty - \beta U_\infty \frac{y - y_c}{R} \exp\left(-\frac{r^2}{2}\right), \\ V_0 &= \beta U_\infty \frac{x - x_c}{R} \exp\left(-\frac{r^2}{2}\right), \\ T_0 &= T_\infty - \frac{\beta U_\infty^2}{2C_p} \exp\left(-\frac{r^2}{2}\right), \end{aligned} \tag{58}$$

where $r := \sqrt{(x - x_c)^2 + (y - y_c)^2}/R$, $R = 0.05$, and $(x_c, y_c) = (0.05, 0.05)$ is the initial position of the vortex center. The Mach number is set to 0.5, $\beta = 0.2$, $T_\infty = 300\,^\circ$K, $P_\infty = 10^5$ N/m^2, $C_p = R_{\text{gas}}\gamma/(\gamma - 1)$, R_{gas} is the ideal gas constant and $\gamma = 7/5$. The reminding variables, ρ and e, are determined by the equation of state for perfect gas. Periodic boundary conditions are used in all dimensions. On a finite domain with periodic boundary conditions, the motion of the vortex is periodic with the period $T = L_x/U_\infty$, where L_x is the domain size in x direction. A uniform mesh of size $[N_x, N_y] = [24, 24]$ is used on a computational domain of size $[0, L_x] \times [0, L_y]$ and $L_x = L_y = 0.1$.

Evaluation of the temporal order of accuracy requires the time-exact solution, which may be approximated by a solution obtained with a sufficiently small time-step size (*i.e.*, we use CFL $= 0.1$) so that the temporal error is negligible. Specifically, the time step size Δt is decreased until the following entropy error becomes a constant

$$E_{L_2(\Omega)}(s) := \sqrt{\frac{1}{|\Omega|}\int_\Omega \left(\frac{s}{s_0}\right)^2 \mathrm{d}\boldsymbol{x} - 1} \tag{59}$$

where $s = p/\rho^\gamma$ and s_0 is the entropy of the free stream.

We use CFL $= 0.1 \times 2^n$ with $0 \le n \le 5$, to measure the order of convergence with respect to the time-exact solution. The order of convergence for the TVDRK3, BDF2, and PCEXP schemes are all shown in Fig. 5(a) with the order of polynomials $p = 0$ to $p = 3$. The formal orders of accuracy are verified for all the cases, which validate our implementation. It is also apparent that the temporal error of BDF2 is much larger than that of PCEXP with an equal time-step size. Overall, the error of PCEXP is one order of magnitude smaller than that of BDF2, although both schemes are second-order accurate.

Besides the accuracy, the computational efficiency is also investigated by showing the evolution of the error with respect to CPU time. We focus on two cases, one with uniform grids, and the other with highly stretched grids.

With uniform meshes, small time-steps are used to test the order of accuracy. Obviously, when the time step size is sufficiently small, both the implicit and exponential schemes are not as efficient as the explicit TVDRK3 scheme because of its simplicity. Therefore, we only compare the CPU times of the exponential and implicit schemes in Fig. 5(b). The results show that the temporal error of the PCEXP scheme not only is one order of magnitude smaller than that of the BDF2 scheme, as shown in Fig. 5(a), but also decays much faster than that of the BDF2 scheme, as shown in Fig. 5(b). This shows that the PCEXP scheme is more accurate and efficient than the BDF2 scheme.

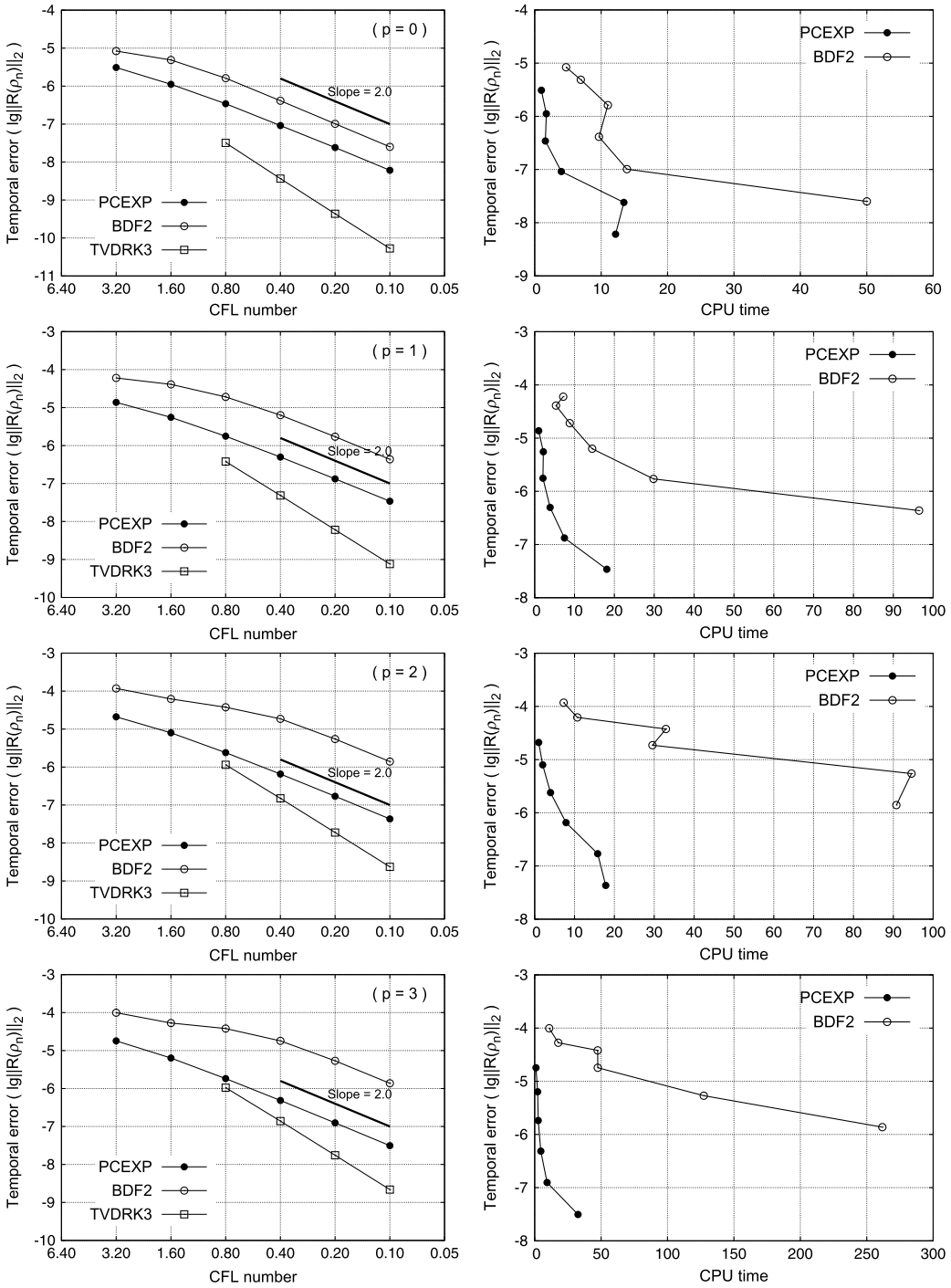
For a stiff case, the following highly stretched non-uniform mesh is used:

$$x_j = \frac{1}{2}\left(1 - \xi_j^3\right) x_{\text{L}} + \frac{1}{2}\left(1 + \xi_j^3\right) x_{\text{R}}, \quad \xi_j = \frac{2}{N}(j - 1) - 1, \quad 1 \le j \le N, \tag{60}$$

where $x_{\text{L}} = 0$, $x_{\text{R}} = 0.1$ and $N = N_x = N_y = 24$. The mesh in y direction is also clustered in the same manner. The grids are concentrated about the cube center, with a minimal grid size of 3×10^{-5} and a maximal one of 0.0115 for inducing the mesh stiffness.

The solution computed by using the third-order TVDRK3 scheme with CFL $= 1.2$ is used as the reference solution. The results computed by using both second-order schemes BDF2 and PCEXP with CFL $= 1000$ are compared with each other since both permit large time steps and should be more efficient in this case. First, the accuracy of the solutions obtained by using both schemes at the end of one period are shown in Fig. 6. The result of BDF2 exhibits a visible phase delay caused by a large temporal error, which is evident in Fig. 5(a), while the result of PCEXP shows very little phase error and agrees well with the result of TVDRK3. This validates the fact that the PCEXP scheme generates a temporal error much smaller than what the BDF2 scheme does.

The total error, which includes both temporal and spatial errors, is an important factor which should be considered. The behavior of the total error versus time is shown in Fig. 7. The errors in the solutions obtained by using the third-order TVDRK3 scheme with CFL $= 1.2$ and 0.1 are also included, and the latter is used as the approximated time-exact solution. Two observations can be made. First, the total errors of the TVDRK3 scheme with CFL $= 1.2$ and 0.1 are almost indistinguishable with a fixed polynomial order p. This suggests that the total error is indeed dominated by spatial error, and the temporal error has little effect, if any. Therefore, the total error will not be reduced by either decreasing the CFL number or using



(a) Convergence order (b) CPU time

Fig. 5. Vortex transportation on a 24×24 uniform mesh with $\mathrm{CFL} = 0.1 \times 2^n$, $0 \le n \le 5$. Left: Temporal convergence of PCEXP, BDF2, and TVDRK3 schemes. Right: Evolution of the errors versus CPU time for PCEXP and BDF2 schemes. From top to bottom: $0 \le p \le 3$. The CPU time is dimensionless.

even higher-order time discretizations. Second, the errors of the PCEXP scheme with $\mathrm{CFL} = 1000$ are rather close to that of the TVDRK3 scheme for all cases of $0 \le p \le 3$; and the PCEXP scheme is certainly more accurate than the BDF2 scheme with the same CFL number.

Finally, the computational efficiency of three schemes, PCEXP, BDF2, and TVDRK3 are compared by measuring the CPU time, and the results are summarized in Table 2. For $p = 0$, the PCEXP scheme is about 3.9 times faster than BDF2, and both the PCEXP and BDF2 schemes are much faster than TVDRK3 while the accuracy is maintained.

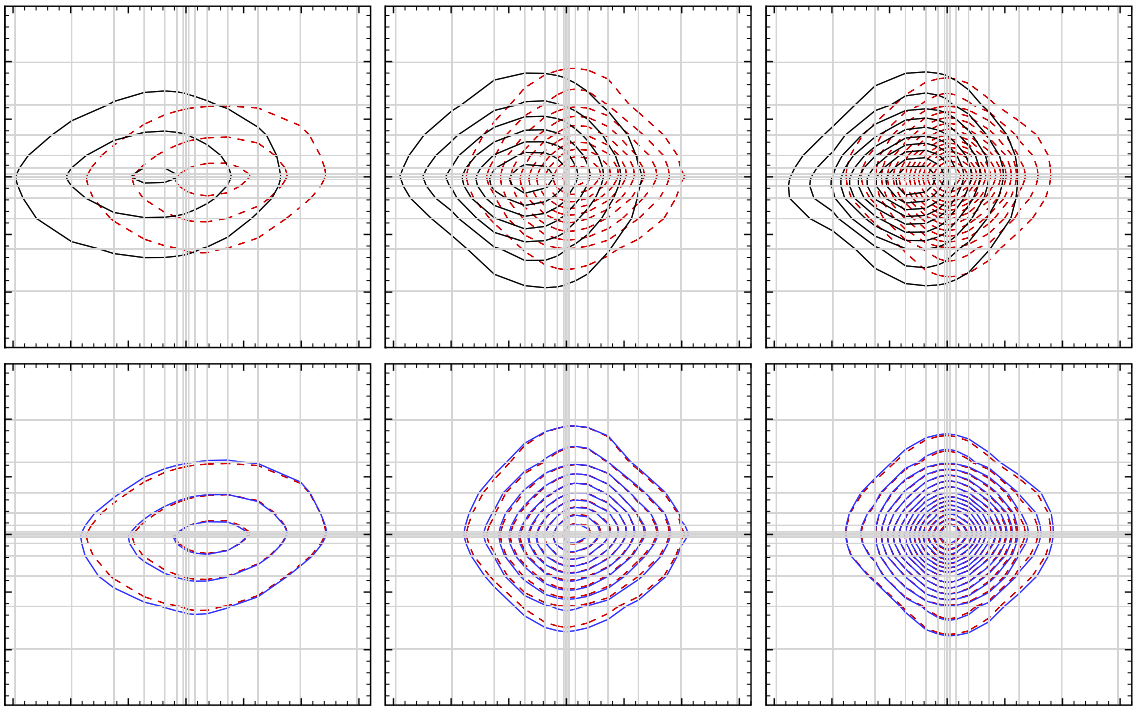


Fig. 6. Vortex transport on a 24×24 stretched mesh. Density Contours of 15 iso-lines in the range of [0.99691, 0.99974] after one period in time. Top: BDF2 (black solid lines) *versus* TVDRK3 (red dashed lines). Bottom: PCEXP (blue solid lines) *versus* TVDRK3 (red dashed lines). From the left to the right: $p = 1$, 2, and 3. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

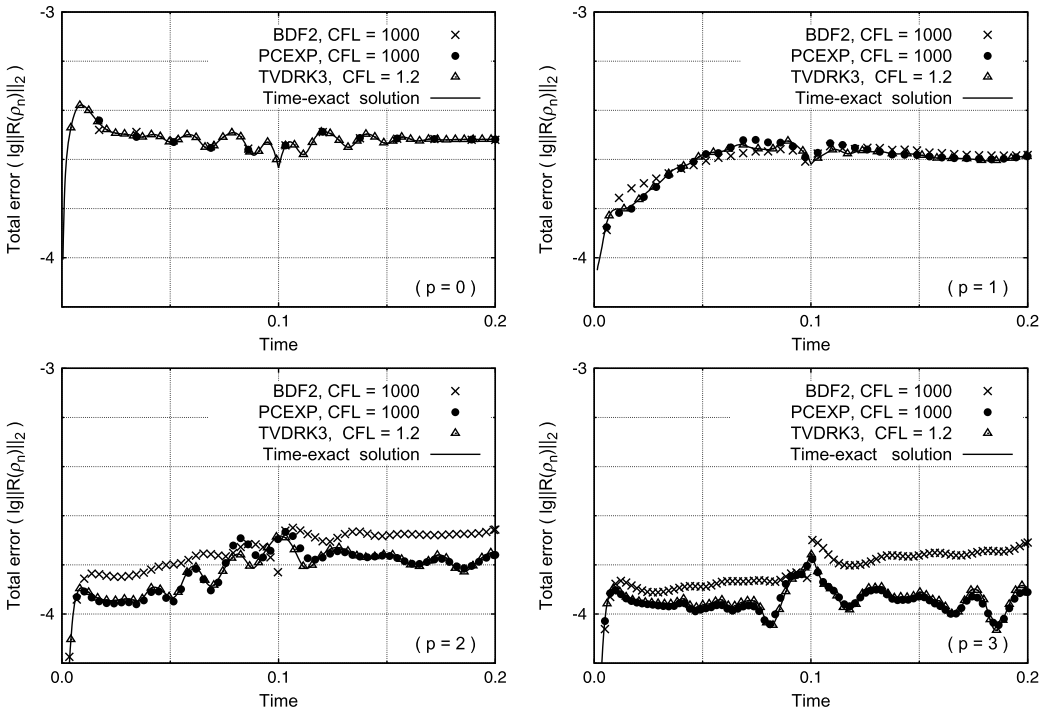


Fig. 7. Vortex transport on a 24×24 uniform mesh. The evolution of the total error of density. The time-exact solution is obtained by using the TVDRK3 scheme with CFL $= 0.1$. The time is dimensionless.

Table 2
Vortex transportation on 24×24 stretched mesh in one period. The t_P, t_B, and t_T are the CPU times (in minutes) corresponding to the PCEXP, BDF2, and TVDRK3 schemes, respectively.

PCEXP				BDF2			TVDRK3		
p order	Steps	CFL	t_P	Steps	CFL	t_B/t_P	Steps	CFL	t_T/t_P
$p=0$	12	1000.0	0.03	12	1000.0	3.90	9696	1.2	48.97
$p=1$	35	1000.0	1.10	35	1000.0	1.13	29106	1.2	10.70
$p=2$	52	1000.0	12.10	52	1000.0	1.00	48507	1.2	4.98
$p=3$	84	1000.0	76.50	84	1000.0	1.60	67893	1.2	3.80

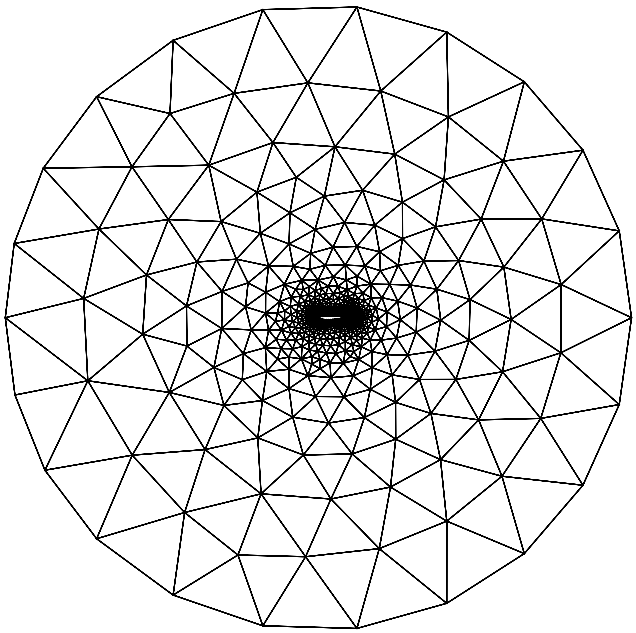


Fig. 8. The subsonic flow over a NACA0012 airfoil with $\mathrm{Ma}=0.63$ and $\alpha=2^\circ$. An illustration of a typical mesh.

5.2. Performance assessments for steady problems

Unconditional stable implicit methods are highly efficient for solving steady problems, often achieving orders of magnitude speedup relative to explicit methods such as the Runge–Kutta types. In this section, the exponential schemes are compared with two implicit methods including the backward Euler (BE) and the second-order backward difference formula (BDF2). Both schemes use an ILU preconditioned GMRES linear solver. Two exponential schemes, the first-order PCEXP scheme, *i.e.*, the EXP1 scheme, which skips the second stage evaluation in (12b) and the second-order PCEXP scheme are applied to steady flow problems in both 2D and 3D. To enhance the computational efficiency and maintain stability for steady problems, the CFL number for all the exponential and implicit schemes is dynamically determined by the following formula:

$$\mathrm{CFL}(n) = \min\left\{\mathrm{CFL}_{\max},\ \max\left[\|R(\rho_n)\|_2^{-3},\ 1+\frac{(n-1)}{(2p+1)}\right]\right\}, \tag{61a}$$

$$\|R(\rho_n)\|_2 := \frac{1}{|\Omega|}\left[\int_\Omega R(\rho_n)^2 \mathrm{d}\boldsymbol{x}\right]^{1/2}, \tag{61b}$$

where $R(\rho_n)$ denotes the residual of density ρ, $\mathrm{CFL}_{\max}$ is the user-defined maximal CFL number, n is the number of iterations, and p is the spatial order of accuracy. Thus, the value of CFL starts at unity initially ($n=1$) when $\|R(\rho_n)\|_2$ is large, and gradually increases to its maximum $\mathrm{CFL}_{\max}$ as $\|R(\rho_n)\|_2$ diminishes.

5.2.1. Subsonic flow over a NACA0012 airfoil in 2D

In this Section, we consider a subsonic flow over the NACA0012 airfoil with the Mach number $\mathrm{Ma}=0.63$ and the angle of attack $\alpha=2^\circ$. The computational domain is a circular disc with the radius of 5 in the unit of the chord length equal to 1, as shown in Fig. 8. The mesh is a quasi-2D one consisting of 1322 quadratic curved wedge elements. The minimal and maximal grid sizes are about 0.01 and 2.0, respectively. The CFL number is determined by (61a) with $\mathrm{CFL}_{\max}=1000$.

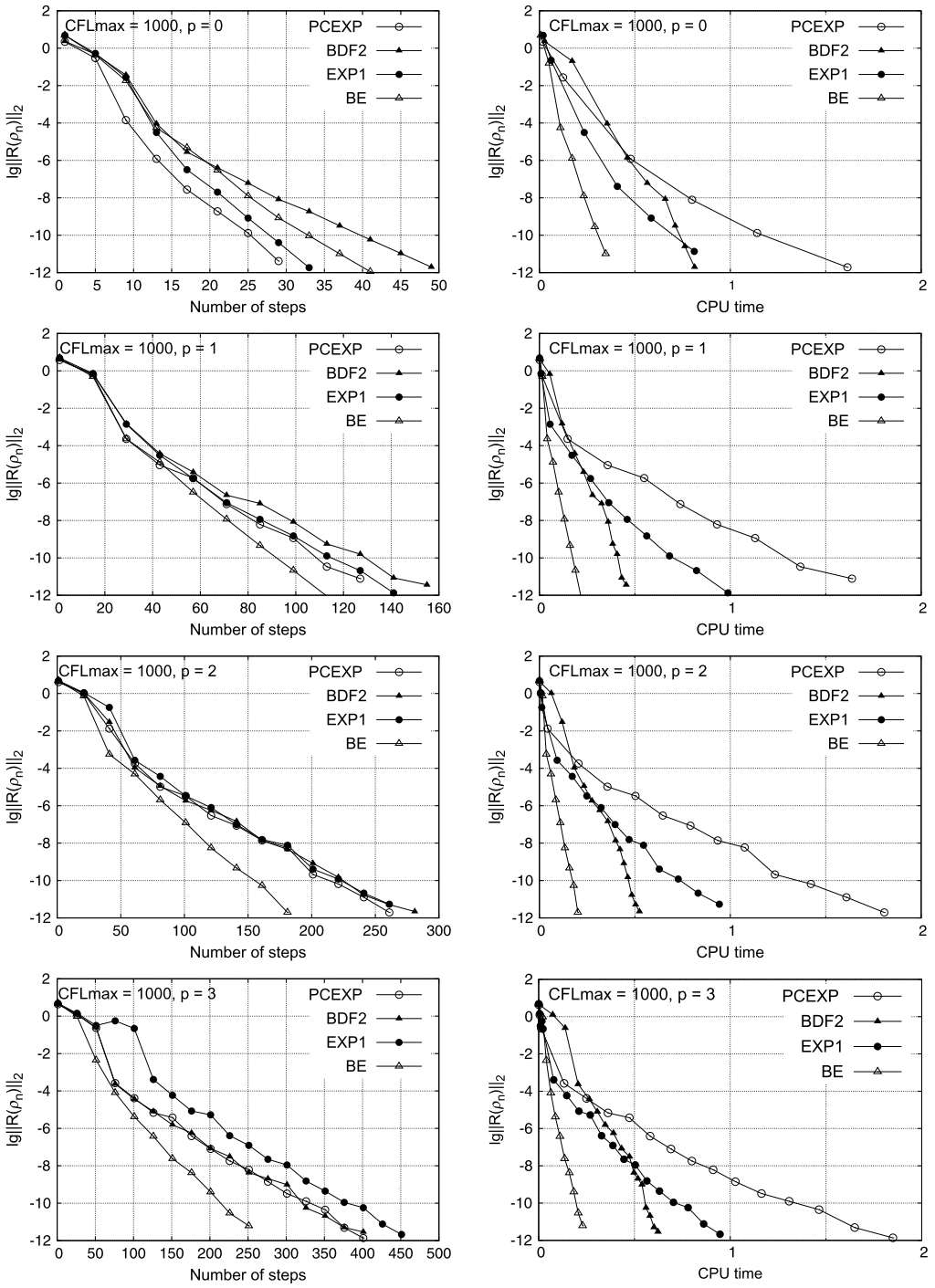


Fig. 9. Flow past a NACA0012 airfoil at Ma = 0.63. Convergence behaviors of the density residual $\lg \|R(\rho_n)\|_2$ for the exponential and implicit schemes in terms of the number of iterations (left) and the elapsed CPU time normalized by that of EXP1 (right), with DG discretizations of $p = 0$ to $p = 3$ (from top to bottom).

Fig. 9 shows the convergence behaviors of the density residual $R(\rho_n)$ with the L_2 norm for all the time-marching schemes with different order p in terms of the number of iterations and CPU time. For $p = 0$, the PCEXP scheme requires the least number of iterations to converge to the steady state, while the BE scheme requires the shortest CPU time. For $1 \leq p \leq 3$, the BE scheme is the most efficient one in terms of both the number of iterations and CPU time. While the number of iterations to attain convergence is rather similar for all the schemes, the required CPU time is rather different; the BE scheme is by far the fastest one in this case.

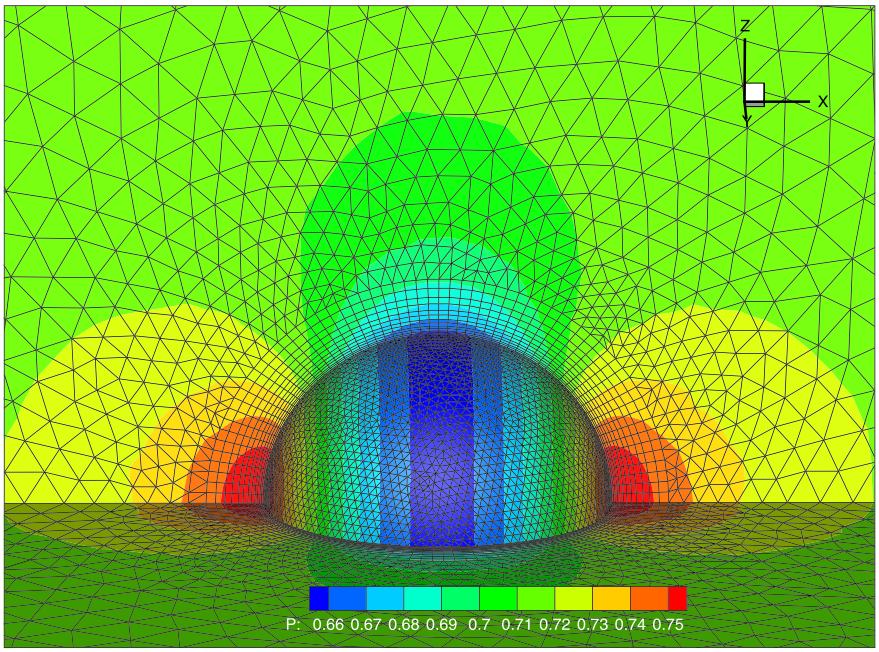


Fig. 10. Flow past a sphere at Ma = 0.3. A close-up view of the mesh of hybrid quadratic curved elements about the sphere along with the pressure computed with $p = 2$ discretization.

5.2.2. Subsonic flow over a sphere in 3D

In this Section, we evaluate the computational efficiency of the exponential schemes for a three-dimensional flow past a sphere with the Mach number Ma = 0.3. The radius of the sphere is set to 1. The computational domain is the spherical shell with the inner and outer radius of 1 and 5, respectively. The inner boundaries of the computational domain are the slip wall, and the outer ones are the far-field characteristic boundaries defined by Riemann invariants. The CFL number of all the schemes is determined by (61a) with $\mathrm{CFL}_{\max} = 1000$.

The mesh respects the flow symmetries of the horizontal and vertical planes, on which the symmetry boundary condition is imposed. The curved mesh consists of 9778 tetrahedrons and 4248 prisms. A close-up view of the mesh about the sphere and the pressure field computed with the PCEXP scheme of $p = 2$ discretization is illustrated in Fig. 10.

Fig. 11 shows the convergence histories of the exponential schemes, EXP1 and PCEXP, and the implicit schemes, BE and BDF2, with the spatial orders $0 \leq p \leq 2$.

The rates of convergence for the exponential and the implicit schemes in terms of the number of iterations are similar, as shown in Fig. 11 (left), because both types of schemes are of global nature. The PCEXP scheme requires the least number of iterations to converge when $p = 0$, while the BE scheme does so when $p = 1$ and 2. We note that the implicit schemes with the GMRES linear solver preconditioned by an ILU is among the fastest solvers. Using other less efficient linear solvers would degrade the efficiency of an implicit solver.

In Fig. 11 (right), the computational efficiency is measured by the elapsed CPU time to achieve convergence. Both first-order schemes, EXP1 and BE, converge faster than their second-order counterparts, PCEXP and BDF2, respectively. Again the BE scheme is the fastest in terms of CPU time, followed closely by the EXP1 scheme. Interestingly, the BDF2 scheme is the slowest in terms of CPU time in this case. It is also observed that the EXP1 scheme converges roughly twice as fast as its second-order counterpart, the PCEXP scheme. This suggests that first-order schemes are the most efficient for steady-state calculations, and higher-order temporal accuracy is inefficient. It can also be seen that the EXP1 scheme performs better than the BDF2 scheme in all cases of $0 \leq p \leq 3$, as opposed to the previous case of the flow past a NACA0012 airfoil in 2D, in which the BDF2 scheme performs better (cf. Fig. 9). For steady problems in 3D, the computational efficiency and performance of the exponential schemes are comparable to those of the implicit schemes in terms of either the number of iterations or CPU time; and the first-order schemes perform better than their second-order counterparts.

6. Conclusions

An exponential scheme, PCEXP, has been developed for the time marching of steady and unsteady inviscid flows in both 2D and 3D. The PCEXP scheme, a one-step scheme based on the predictor–corrector methodology, allows large time-step size while maintaining second-order accuracy in time. The temporal accuracy of the PCEXP scheme is verified; its convergence behavior and computational efficiency are validated for both steady and unsteady test cases.

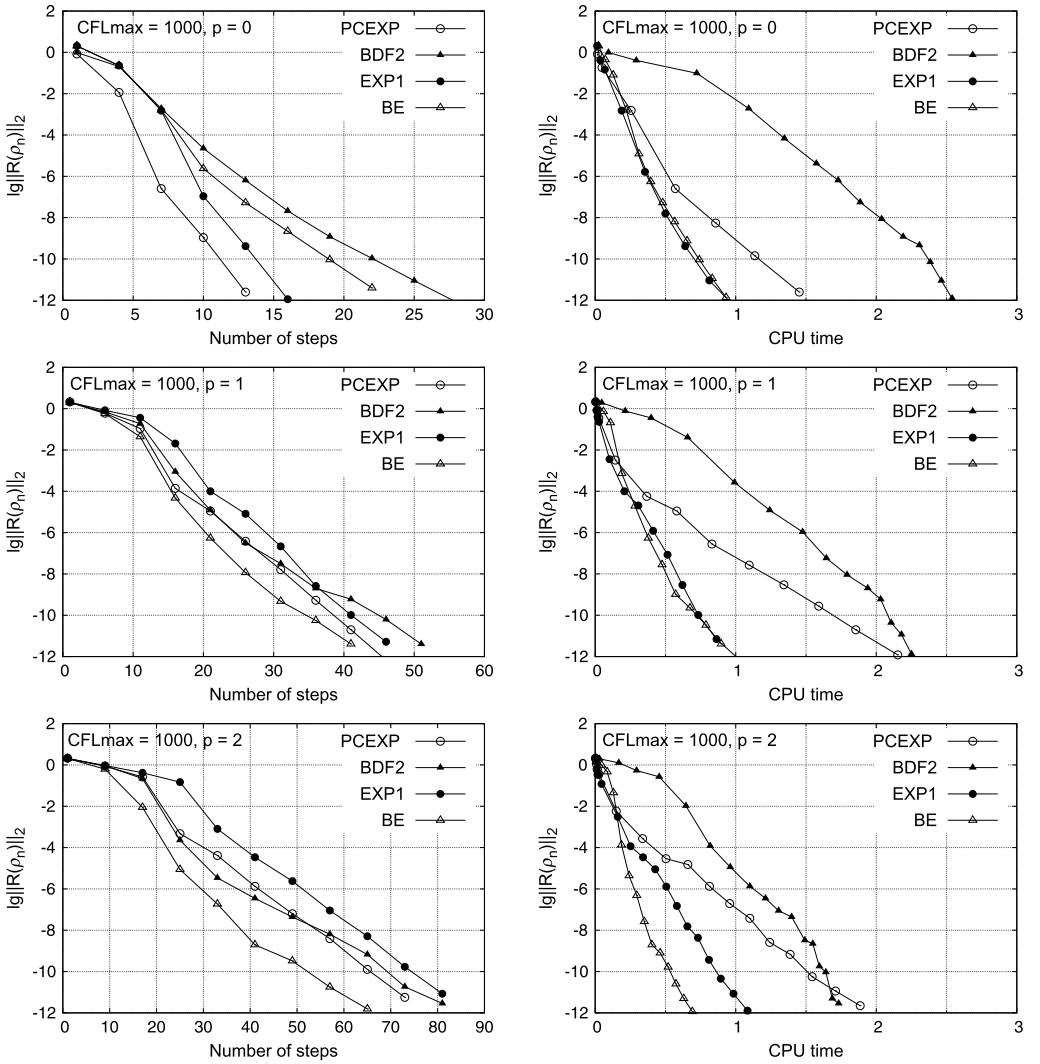


Fig. 11. The subsonic flow over a sphere in 3D with Ma = 0.3. Convergence behaviors of the density residual $\lg \|R(\rho_n)\|_2$ for the exponential and implicit schemes in terms of the number of iterations or steps (left) and the elapsed CPU time normalized by that of EXP1 (right), with DG discretizations of $p = 0$ to $p = 2$ (from top to bottom).

The unsteady flow of a vortex moving with a constant velocity in 2D is used to verify the temporal accuracy of the proposed PCEXP scheme. The comparisons are carried out with uniform (nonstiff case) and highly clustered non-uniform (stiff case) meshes. On the uniform meshes, the order of temporal accuracy of the PCEXP and BDF2 schemes are verified. In addition, the PCEXP scheme is shown to be much more accurate than the BDF2 scheme. Specifically for the vortex transportation in 2D, the magnitude of total error in the solution of the PCEXP scheme can be more than one order of magnitude smaller than that of the BDF2 scheme with the same time-step size. Thus, the PCEXP scheme is far more effective and more efficient than the BDF2 scheme. The above conclusion applies to both nonstiff and stiff cases.

For steady-state problems, the PCEXP scheme allows large time-step sizes thus can achieve a rapid convergence. Both the EXP1 and PCEXP schemes enjoy the rates of convergence comparable to their implicit counterparts, the BE and BDF2 schemes, respectively, in terms of the number of iterations. However, it should be noted that, in terms of CPU time, the BE scheme is in fact faster than the PCEXP scheme for steady-state problems. Also, the first-order exponential scheme, EXP1, is more efficient than its second-order counterpart, PCEXP, as expected.

In conclusion, we have successfully demonstrated the effectiveness and efficiency of the proposed PCEXP scheme for accelerating computations of unsteady flows, especially for stiff problems. Comparing to the BDF2 scheme, the PCEXP scheme generates a much smaller temporal error, although both schemes are second-order accurate. To enhance the efficiency of exponential schemes including the PCEXP scheme, the computational cost per iteration of the matrix exponential worth a further investigation.

Acknowledgements

This work is funded by the National Natural Science Foundation of China (NSFC) under the Grant U1530401. The computational resources are provided by the Special Program for Applied Research on Super Computing from the NSFC–Guangdong Joint Fund (Phase 2) under Grant U1501501 and Beijing Computational Science Research Center (CSRC). L. Ju would like to acknowledge the support from the US National Science Foundation under the Grant DMS-1521965 and the U.S. Department of Energy under the Grant DE-SC0016540. The authors would like to thank Dr. Ken C.Y. Loh for his careful proof-reading of the manuscript. The authors would also like to thank the anonymous referees whose comments helped us improve the paper significantly.

Appendix A. The Jacobian matrices

The matrix $\nabla\psi\,\partial\mathbf{F}/\partial\mathbf{U}$ in (50) is

$$\begin{pmatrix} -B_2 & \psi_x & \psi_y & \psi_z & 0 \\ a_0\psi_x - B_1 u & B_1 - B_2 - a_3 u\psi_x & u\psi_y - a_2 v\psi_x & u\psi_z - a_2 w\psi_x & a_2\psi_x \\ a_0\psi_y - B_1 v & v\psi_x - a_2 u\psi_y & B_1 - B_2 - a_3 v\psi_y & v\psi_z - a_2 w\psi_y & a_2\psi_y \\ a_0\psi_z - B_1 w & w\psi_x - a_2 u\psi_z & w\psi_y - a_2 v\psi_z & B_1 - B_2 - a_3 w\psi_z & a_2\psi_z \\ (a_0 - a_1)B_1 & a_1\psi_x - a_2 B_1 u & a_1\psi_y - a_2 v B_1 & a_1\psi_z - a_2 B_1 w & \gamma B_1 - B_2 \end{pmatrix}, \tag{A.1}$$

where $\boldsymbol{v} := (u,\, v,\, w)$, $\boldsymbol{\omega} := (\omega_x,\, \omega_y,\, \omega_z)$, $\nabla\psi := (\psi_x,\, \psi_y,\, \psi_z)$,

$$\begin{aligned} &a_0 := \frac{1}{2}(\gamma - 1)(u^2 + v^2 + w^2), \quad a_1 := \gamma e - a_0, \quad a_2 := \gamma - 1, \quad a_3 := \gamma - 2, \\ &B_1 := \boldsymbol{v}\cdot\nabla\psi = u\psi_x + v\psi_y + w\psi_z, \quad B_2 := (\boldsymbol{\omega}\times\boldsymbol{x})\cdot\nabla\psi. \end{aligned} \tag{A.2}$$

The source-term Jacobian matrix $\partial\mathbf{S}/\partial\mathbf{U}$ in (50) is

$$\frac{\partial\mathbf{S}}{\partial\mathbf{U}} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\omega_z & \omega_y & 0 \\ 0 & \omega_z & 0 & -\omega_x & 0 \\ 0 & -\omega_y & \omega_x & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}. \tag{A.3}$$

References

[1] W. Reed, T. Hill, Anisotropic Refinement Algorithms for Finite Elements, Tech. Rep. LA-UR-73-479, Los Alamos National Laboratories, 1973.
[2] P. Lesaint, P. Raviart, On a finite element method for solving the neutron transport equation, in: Mathematical Aspects of Finite Elements in Partial Differential Equations, Academic Press, New York, 1974, pp. 89–123.
[3] B. Cockburn, C.-W. Shu, TVB Runge–Kutta local projection discontinuous Galerkin finite element method for conservation laws II. General framework, Math. Comput. 52 (186) (1989) 411–435.
[4] K. Bey, J. Oden, A Runge–Kutta Discontinuous Finite Element Method for High Speed Flows, Tech. Rep. AIAA-1991-1575, AIAA, 1991.
[5] F. Bassi, S. Rebay, High-order accurate discontinuous finite element solution of the 2D Euler equations, J. Comput. Phys. 138 (2) (1997) 251–285.
[6] B. Cockburn, C.-W. Shu, The Runge–Kutta discontinuous Galerkin method for conservation laws V. Multidimensional systems, J. Comput. Phys. 141 (2) (1998) 199–224.
[7] B. Cockburn, C.-W. Shu, The local discontinuous Galerkin method for time-dependent convection–diffusion systems, SIAM J. Numer. Anal. 35 (6) (1998) 2440–2463.
[8] B. Cockburn, G. Karniadakis, C.-W. Shu (Eds.), Discontinuous Galerkin Methods: Theory, Computation and Applications, Lecture Notes in Computational Science and Engineering, vol. 11, Springer, New York, 2000.
[9] S. Ii, F. Xiao, High order multi-moment constrained finite volume method. Part I: basic formulation, J. Comput. Phys. 228 (10) (2009) 3669–3707.
[10] H. Huynh, A Flux Reconstruction Approach to High-Order Schemes Including Discontinuous Galerkin Methods, Tech. Rep. AIAA-2007-4079, AIAA, 2012.
[11] Z.J. Wang, H. Gao, A unifying lifting collocation penalty formulation including the discontinuous Galerkin, spectral volume/difference methods for conservation laws on mixed grids, J. Comput. Phys. 228 (21) (2009) 8161–8186.
[12] H. Huynh, Z.J. Wang, P.E. Vincent, High-order methods for computational fluid dynamics: a brief review of compact differential formulations on unstructured grids, Comput. Fluids 98 (2014) 209–220.
[13] C.-W. Shu, An overview on high order numerical methods for convection dominated PDEs, in: T.Y. Hou, E. Tadmor (Eds.), Hyperbolic Problems: Theory, Numerics, Applications, Springer, Berlin, 2002, pp. 79–88.
[14] Z.J. Wang, K. Fidkowski, R. Abgrall, F. Bassi, D. Caraeni, A. Cary, H. Deconinck, R. Hartmann, K. Hillewaert, H. Huynh, N. Kroll, G. May, P.-O. Persson, B. van Leer, M. Visbal, High-order CFD methods: current status and perspective, Int. J. Numer. Methods Fluids 72 (8) (2013) 811–845.
[15] Z.J. Wang, H. Huynh, A review of flux reconstruction or correction procedure via reconstruction method for the Navier–Stokes equations, Mech. Eng. Rev. 3 (1) (2016) 15-00475.
[16] S. Cox, P. Matthews, Exponential time differencing for stiff systems, J. Comput. Phys. 176 (2) (2002) 430–455.
[17] M. Hochbruck, C. Lubich, On Krylov subspace approximations to the matrix exponential operator, SIAM J. Numer. Anal. 34 (5) (1997) 1911–1925.
[18] M. Hochbruck, C. Lubich, H. Selhofer, Exponential integrators for large systems of differential equations, SIAM J. Sci. Comput. 19 (5) (1998) 1552–1574.
[19] A. Ostermann, M. Thalhammer, W. Wright, A class of explicit exponential general linear methods, BIT Numer. Math. 46 (2) (2006) 409–431.
[20] M. Tokman, Efficient integration of large stiff systems of ODEs with exponential propagation iterative (EPI) methods, J. Comput. Phys. 213 (2) (2006) 748–776.
[21] Q. Nie, Y.-T. Zhang, R. Zhao, Efficient semi-implicit schemes for stiff systems, J. Comput. Phys. 214 (2) (2006) 521–537.

[22] Q. Nie, F. Wan, Y.-T. Zhang, X.-F. Liu, Compact integration factor methods in high spatial dimensions, J. Comput. Phys. 227 (10) (2008) 5238–5255.
[23] S. Chen, Y.-T. Zhang, Krylov implicit integration factor methods for spatial discretization on high dimensional unstructured meshes: application to discontinuous Galerkin methods, J. Comput. Phys. 230 (11) (2011) 4336–4352.
[24] M. Caliari, A. Ostermann, Implementation of exponential Rosenbrock-type integrators, Appl. Numer. Math. 59 (3) (2009) 568–582.
[25] M. Hochbruck, A. Ostermann, J. Schwitzer, Exponential Rosenbrock-type methods, SIAM J. Numer. Anal. 47 (1) (2009) 786–803.
[26] M. Tokman, A new class of exponential propagation iterative methods of Runge–Kutta type (EPIRK), J. Comput. Phys. 230 (24) (2011) 8762–8778.
[27] J. Loffeld, M. Tokman, Comparative performance of exponential, implicit, and explicit integrators for stiff systems of ODEs, J. Comput. Appl. Math. 241 (2013) 45–67.
[28] L. Ju, L. Zhu, J. Zhang, Q. Du, Fast explicit integration factor methods for semilinear parabolic equations, J. Sci. Comput. 62 (2) (2015) 431–455.
[29] L. Zhu, L. Ju, W. Zhao, Fast high-order compact exponential time differencing Runge–Kutta methods for second-order semilinear parabolic equations, J. Sci. Comput. 67 (3) (2016) 1043–1065.
[30] W. Edwards, L. Tuckerman, R. Friesner, D. Sorensen, Krylov method for the incompressible Navier–Stokes equations, J. Comput. Phys. 110 (1) (1994) 82–102.
[31] J. Schulze, P. Schmid, J. Sesterhenn, Exponential time integration using Krylov subspaces, Int. J. Numer. Methods Fluids 60 (2009) 591–609.
[32] C. Clancy, J. Pudykiewicz, On the use of exponential time integration methods in atmospheric models, Tellus A 65 (2013) 20898.
[33] M. Tokman, J. Loffeld, Efficient design of exponential-Krylov integrators for large scale computing, Proc. Comput. Sci. 1 (1) (2010) 229–237.
[34] Y. Saad, Analysis of some Krylov subspace approximations to the matrix exponential operator, SIAM J. Numer. Anal. 29 (1) (1992) 209–228.
[35] Y. Saad, Iterative Methods for Sparse Linear Systems, 2nd edition, SIAM, Philadelphia, 2003.
[36] C. Moler, C. van Loan, Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later, SIAM J. Numer. Anal. 45 (1) (2003) 3–49.
[37] E. Toro, Riemann Solvers and Numerical Methods for Fluid Dynamics, Springer, Berlin, 1999.
[38] P. Roe, Approximate Riemann solvers, parameter vectors, and difference schemes, J. Comput. Phys. 43 (2) (1981) 357–372.
[39] S. Deng, W. Cai, Analysis and application of an orthogonal nodal basis on triangles for discontinuous spectral element methods, Appl. Numer. Anal. Comput. Math. 2 (3) (2006) 326–345.
[40] M. Bergot, M. Durufl, Higher order discontinuous Galerkin method for pyramidal elements using orthogonal bases, Numer. Methods Partial Differ. Equ. 29 (1) (2013) 144–169.
[41] L. Botti, Influence of reference-to-physical frame mappings on approximation properties of discontinuous piecewise polynomial spaces, J. Sci. Comput. 52 (3) (2012) 675–703.
[42] A. Wolkov, C. Hirsch, N. Petrovskaya, Application of a higher order discontinuous Galerkin method in computational aerodynamics, Math. Model. Nat. Phenom. 6 (3) (2011) 237–263.
[43] H. Luo, S. Li, Y. Xia, R. Nourgaliev, C. Cai, A Hermit WENO Reconstruction-Based Discontinuous Galerkin Method for the Euler Equations on Tetrahedral Grids, Tech. Rep. AIAA-2012-0461, AIAA, 2012.
[44] S. Eckert, H. Baaser, D. Gross, O. Scherf, A BDF2 integration method with step size control for elasto-plasticity, Comput. Mech. 34 (5) (2004) 377–386.
[45] CFL3D (version 5.0) user's manual, http://cfl3d.larc.nasa.gov, 2016.