Journal of Scientific Computing manuscript No.
(will be inserted by the editor)

Optimal strong-stability-preserving Runge–Kutta time discretizations for discontinuous Galerkin methods

Ethan J. Kubatko · Benjamin A. Yeager · David I. Ketcheson

Received: date / Accepted: date

Abstract Discontinuous Galerkin (DG) spatial discretizations are often used in a method-of-lines approach with explicit strong-stability-preserving (SSP) Runge–Kutta (RK) time steppers for the numerical solution of hyperbolic conservation laws. The time steps that are employed in this type of approach must satisfy Courant–Friedrichs–Lewy (CFL) stability constraints that are dependent on both the region of absolute stability and the SSP coefficient of the RK method. While existing SSPRK methods have been optimized with respect to the latter, it is in fact the former that gives rise to stricter constraints on the time step in the case of RKDG stability. Therefore, in this work, we present the development of new "DG-optimized" SSPRK methods with stability regions that have been specifically designed to maximize the stable time step size for RKDG methods of a given order in one space dimension. These new methods represent the best available RKDG methods in terms of computational efficiency, with significant improvements over methods using existing SSPRK time steppers that have been optimized with respect to SSP coefficients. Second-, third-, and fourth-order methods with up to eight stages are presented, and their stability properties are verified through application to numerical test cases.

Keywords discontinuous Galerkin · Runge–Kutta · strong-stability-preserving

E. Kubatko
Department of Civil, Environmental, and Geodetic Engineering, The Ohio State University, 2070 Neil Avenue, Columbus, OH 43210, United States
Tel.: 614-292-7176
E-mail: kubatko.3@osu.edu

B. Yeager
Department of Civil, Environmental, and Geodetic Engineering, The Ohio State University, 2070 Neil Avenue, Columbus, OH 43210, United States

D. Ketcheson
Division of Mathematical and Computer Sciences and Engineering, 4700 King Abdullah University of Science & Technology, Thuwal 23955, Saudi Arabia

1 Introduction

Many physical problems take the form of time-dependent hyperbolic conservation laws — examples include inviscid shallow water flow, the propagation of waves in elastic solids, gas dynamics, traffic flow, etc. These problems are often solved using a method-of-lines approach, where discretization of the spatial operators of the hyperbolic partial differential equations (PDEs) yields a set of time-dependent ordinary differential equations (ODEs) that must then be integrated in time. Among the various spatial discretization methods used for hyperbolic problems, discontinuous Galerkin (DG) finite element methods have become a particularly popular choice and have been widely used in a broad range of applications; see, for example, [2,21,23,30,32,35,36]. The system of time-dependent ODEs resulting from the DG spatial discretization is often integrated in time using explicit Runge–Kutta (RK) methods. This approach gives rise to the RKDG methods originally developed by Cockburn and Shu; see the review article [5].

As is well known, when using explicit time discretization methods, the time steps that are employed must satisfy Courant–Friedrichs–Lewy (CFL) stability constraints. For hyperbolic problems, these include conditions to maintain both (*i*) linear stability, in order to guarantee convergence of the methods in the case of smooth solutions, and (*ii*) some form of nonlinear stability, such as total variation (TV) stability, in order to control non-physical oscillations of the numerical approximations in the presence of discontinuities or shocks. With respect to the latter property, it can be shown that a method-of-lines approach using DG spatial discretizations in conjunction with the first-order forward Euler method is TVD/TVB (that is, total variation diminishing/bounded) in the means under a suitable CFL restriction, provided a generalized slope limiter is used when higher order (≥ 2) DG spatial discretizations are employed [5]. This nonlinear stability property can be extended to higher order RKDG methods if a special class of RK methods is used — what have come to be called strong-stability-preserving (SSP) RK methods; see, for example, [8], which discusses and demonstrates by example the practical importance of the SSP property.

The defining feature of SSP methods is that they can be written as convex combinations of forward Euler steps. This implies that stability in any norm, seminorm, or convex functional that has been demonstrated with the forward Euler method will carry over to the higher order RK method (as is the case with RKDG methods) under a CFL restriction that depends on the CFL or SSP coefficient of the method (we adopt the latter nomenclature here). There has been a significant amount of research in the area of SSP methods aimed at finding RK methods with optimal SSP coefficients; see, for example, [9,10,11,18,27,28,29]. When paired with higher order DG spatial discretizations these SSPRK methods are "optimal" in the sense that they allow the largest *nonlinearly stable* time step size.

However, higher order DG spatial discretizations coupled with forward Euler time stepping are *linearly unstable* under any constant time step to mesh size ratio, even though the complete RKDG method may be stable; see, for example, [3]. This means that conditions for the linear stability of RKDG methods are not characterized by SSP coefficients (i.e., the main theorem of SSP methods does not apply, see Theorem 2.1 of [8]) but rather by the stability region of the RK methods and the (scaled) spectral radius of the DG spatial operator. For all existing SSPRK methods, the conditions for linear stability of RKDG methods are stricter than those for nonlinear stability (see Table 1 of Section 3), and as demonstrated numerically in [22], and later in this work (see Table 5 of Section 5), it is the linear stability conditions that must be respected in practice or the high-order convergence of the RKDG methods will degenerate to first-order. The main consequence of this is that previously derived SSPRK methods with optimal SSP coefficients are not optimal for DG spatial discretizations, that is, they do not maximize the allowable time step size that will maintain *both* higher order convergence and stability.

Therefore, in this work we outline the construction of new, optimal SSPRK methods with stability regions designed specifically for DG spatial discretization of hyperbolic PDEs in one space dimension. Many authors have investigated the problem of finding stability regions that are optimal in some sense, e.g., stability regions that have maximal extent along the imaginary or real axis [16, 20], those that contain the maximal disk [17, 33], etc; see also [19, 24, 25] and the references therein. The closest work to the current paper is that presented in [31], which sought to derive third- and fourth-order RK methods that are optimal for DG, over a range of spatial orders, considering both accuracy and linear stability (SSP properties were not considered). This work is discussed at greater length in Section 4.2. Here, we construct optimal second-, third-, and fourth-order RKDG methods by deriving new SSPRK methods with both nonlinear and linear stability properties that minimize the amount of computational effort required to reach a given time T.

The rest of this paper is organized as follows. In the next section, we provide a brief overview of the DG spatial discretization of the one-dimensional scalar hyperbolic conservation law. Following this, in Section 3, we provide some pertinent information on RK methods, where we specifically discuss linear stability requirements for RK methods applied to DG spatial discretizations. This is followed by a description of the solution procedure used to construct the new "DG-optimized" SSPRK methods and a presentation of the results. Numerical test cases that demonstrate the efficiency and order of the new methods are presented in Section 5, and finally in Section 6 we draw some conclusions and discuss future directions for this work.

2 The discontinuous Galerkin spatial discretization

We consider a time-dependent hyperbolic conservation law in one space dimension, i.e.,

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}\big(f\left(u\right)\big) = 0 \quad \text{for } x \in \Omega = (a, b), \quad t > 0, \tag{1}$$

with an initial condition $u(x, 0) = u_0(x)$ and periodic boundary conditions $u(a, t) = u(b, t)$. The function $f(u)$ is typically referred to as the flux function, and in the analysis that follows, we consider the linear case, i.e., $f = cu$, where c is a specified constant, which represents the advection, or wave propagation, speed of the problem.

To apply a DG spatial discretization to (1), we begin by introducing a partition of Ω, $a = x_0 < x_1 < \cdots x_N = b$, and set

$$\Omega_j = [x_{j-1}, x_j], \qquad h_j = x_j - x_{j-1}, \quad \text{for } j = 1, 2, \ldots, N.$$

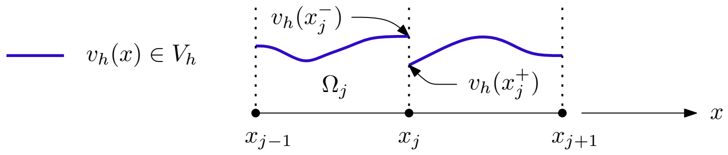

Fig. 1 Illustration of the possible jumps in $v_h \in V_h^p$ at element boundaries.

Given this finite element partition, we can obtain a weak form of the problem by multiplying (1) by a sufficiently smooth test function $v(x)$ and integrating over each element Ω_j, giving

$$\int_{\Omega_j} \frac{\partial u}{\partial t}\, v\, dx - \int_{\Omega_j} f\, \frac{dv}{dx}\, dx + f_j(t)\, v(x_j) - f_{j-1}(t)\, v(x_{j-1}) = 0. \tag{2}$$

We note that the term involving the spatial derivative of f has been integrated by parts, which yields the boundary flux terms denoted by $f_j(t) \equiv f(u(x_j, t))$.

Next, we replace the exact solution u and the test function v of (2) by the functions u_h and v_h, respectively, which belong to the finite-dimensional space of functions chosen as

$$V_h^p = \left\{ v : \ v|_{\Omega_j} \in P^p(\Omega_j) \right\},$$

where $P^p(\Omega_j)$ denotes the space of polynomials over Ω_j of degree $\leq p$. Note that with this choice, functional continuity is not enforced across element boundaries x_{j-1} and x_j, and we indicate the left and right limits of a function $v_h \in V_h^p$ at, e.g., x_j by $v_h(x_j^-)$ and $v_h(x_j^+)$, respectively; see Figure 1. As both u_h and v_h may be discontinuous at element boundaries, we replace the boundary fluxes f_j by appropriately defined numerical fluxes $\widehat{f_j}$, which, in general, depend on both $f(u_h(x_j^-, t))$ and $f(u_h(x_j^+, t))$, and we take the values of the test functions at the boundaries from inside element j. Making these substitutions, we arrive at the following discrete weak form of the problem: find $u_h \in V_h^p$ such that for all test functions $v_h \in V_h^p$ and over each element

$$\int_{\Omega_j} \frac{\partial u_h}{\partial t}\, v_h\, dx - \int_{\Omega_j} f\, \frac{dv_h}{dx}\, dx + \widehat{f}_j(t)\, v_h(x_j^-) - \widehat{f}_{j-1}(t)\, v_h(x_{j-1}^+) = 0. \tag{3}$$

Given a set of basis functions $\boldsymbol{\Phi} = [\,\phi_0, \phi_1, \ldots, \phi_p\,]^\mathsf{T}$ for the space V_h^p, the discrete solution u_h over Ω_j can be expressed as

$$u_h|_{\Omega_j} = \mathbf{u}_j^\mathsf{T}\, \boldsymbol{\Phi}$$

where $\mathbf{u}_j$ is a vector of the (time-dependent) degrees of freedom of the finite element solution over Ω_j. Equation (3) for each Ω_j can then be written as a system of ODEs in the form

$$\mathbf{M}_j \frac{d\mathbf{u}_j}{dt} = \mathbf{F}_j,$$

where the components of the $(p+1) \times (p+1)$ element mass matrix are given by $M_{kl} = \int_{\Omega_j} \phi_k \phi_l \, dx$ and

$$\mathbf{F}_j = \left[\ F_j(\phi_0), F_j(\phi_1), \ldots, F_j(\phi_p)\ \right]^\mathsf{T}$$

with

$$F_j(\phi_i) = \int_{\Omega_j} f \, \frac{d\phi_i}{dx} \, dx + f_{j-1}(t) \, \phi_i(x_{j-1}^+) - f_j(t) \, \phi_i(x_j^-).$$

We note that $\mathbf{M}_j$ is a local (element) matrix and that the elements are coupled through the numerical flux terms appearing in $\mathbf{F}_j$.

Finally, inverting the element mass matrices, the full set of semidiscrete equations over all elements can be written as

$$\frac{d\mathbf{u}}{dt} = \mathbf{L}\left(\mathbf{u}\right), \tag{4}$$

where

$$\mathbf{u} = \left[\; \mathbf{u}_1^\intercal, \mathbf{u}_2^\intercal, \ldots, \mathbf{u}_2^\intercal \; \right]^\intercal,$$

and

$$\mathbf{L} = \left[\; \left(\mathbf{M}_1^{-1}\mathbf{F}_1\right)^\intercal, \left(\mathbf{M}_2^{-1}\mathbf{F}_2\right)^\intercal, \ldots, \left(\mathbf{M}_N^{-1}\mathbf{F}_N\right)^\intercal \; \right]^\intercal.$$

We will refer to $\mathbf{L}$ as the DG spatial operator. In the case of a linear flux function, $f = cu$ with a constant c, $\mathbf{L} = \mathbf{L}_h\mathbf{u}$, where $\mathbf{L}_h$ is a constant matrix dependent on the mesh sizes h_j.

3 The Runge–Kutta time discretization

Prior to considering RK time discretization of the system of ODEs resulting from the DG spatial discretization, consider the application of the forward Euler method to (4), i.e,

$$\mathbf{u}^{n+1} = \mathbf{u}^n + \Delta t \, \mathbf{L}(\mathbf{u}^n), \tag{5}$$

where $\mathbf{u}^n \equiv \mathbf{u}\,(n\Delta t)$, $n \in \mathbb{N}$. With the application of a generalized slope limiter (see, e.g., [4]), this scheme is TVDM (TVD in the means) under the condition

$$\Delta t \;\leq\; \Delta t_{\mathrm{FE}} = \frac{\min_j\left(h_j\right)}{2\left(L_1 + L_2\right)}\,, \tag{6}$$

where L_1 and L_2 are the Lipschitz constants of $\widehat{f}$ with respect to the first and second arguments, respectively; see, for example, [5] for a proof of (6), which makes use of Harten's lemma [14]. In the case of a linear flux function, $f = cu$ with $c > 0$, the use of a simple upwind flux, i.e., $\widehat{f}_j(t) = cu_h(x_j^-, t)$, and a uniform mesh spacing $h_j = \Delta x$, (6) yields the following condition for TV-stability:

$$|c| \, \frac{\Delta t}{\Delta x} \;\leq\; \frac{1}{2}.$$

A von Neumann stability analysis of (5) for the linear case, however, reveals that the scheme is *linearly unstable* under any constant $\Delta t/\Delta x$ when polynomials of degree $p > 0$ are employed; see, for example, [3]. (Note: the $p = 0$ case corresponds to the classic first-order upwind finite difference scheme, which, as is well known, is linearly stable for $|c| \, \Delta t/\Delta x \leq 1$.) To overcome this problem, Cockburn and Shu devised the RKDG methods, which make use of higher order RK methods for the time discretization of (4) when polynomials of degree $p > 0$ are used; see, for example, the review article [5]. In contrast to the case of using the forward

Euler method, these schemes can be shown to be linearly stable under a condition of the form

$$|c| \frac{\Delta t}{\Delta x} \le \mu(p, \mathcal{S}),$$

where μ is a constant that is a function of both the polynomial degree p of the DG spatial discretization and the region of absolute stability $\mathcal{S}$ of the RK method but is independent of the mesh size Δx. For example, an RKDG method using a $p = 1$ DG spatial discretization paired with a two-stage, second-order RK time discretization is stable for $\mu = 1/3$; see [3].

The TV-stability of the RKDG methodology follows readily from the result obtained for the forward Euler case provided SSPRK methods are used. To see this, consider the application of an s-stage RK method to the set of semidiscrete DG equations written in Shu–Osher representation, i.e.,

$$\begin{aligned} \mathbf{u}^{(0)} &= \mathbf{u}^n \\ \mathbf{u}^{(i)} &= \sum_{l=0}^{i-1} \alpha_{il} u^{(l)} + \Delta t\, \beta_{il}\, \mathbf{L}(\mathbf{u}^{(l)}), \quad i = 1, 2, ..., s \\ \mathbf{u}^{n+1} &= \mathbf{u}^{(s)}. \end{aligned} \tag{7}$$

If, in addition to the constraints placed on α_{il} and β_{il} related to order (see, e.g., [1]) and consistency ($\sum_{l=0}^{i-1} \alpha_{il} = 1,\ i = 1, 2, \ldots, s$), the following additional constraints are enforced:

$$\text{(i)}\quad \alpha_{il} \ge 0 \text{ and } \beta_{il} \ge 0, \quad \text{(ii)}\quad \alpha_{il} = 0 \text{ only if } \beta_{il} = 0,$$

then the stages of the RK method can be written as a convex combination of forward Euler steps of sizes $\frac{\beta_{il}}{\alpha_{il}} \Delta t$, i.e.,

$$\mathbf{u}^{(i)} = \sum_{l=0}^{i-1} \alpha_{il} \left[u^{(l)} + \Delta t\, \frac{\beta_{il}}{\alpha_{il}}\, \mathbf{L}(\mathbf{u}^{(l)}) \right], \quad i = 1, 2, ..., s.$$

Written in this way and given (6), it is then easy to see that the resulting RKDG scheme is TVDM under the condition

$$\max_{i,l} \left(\frac{\beta_{il}}{\alpha_{il}} \right) \Delta t \le \Delta t_{\mathrm{FE}},$$

which, again considering the linear case, can be written as

$$|c| \frac{\Delta t}{\Delta x} \le \nu = \frac{1}{2} \underbrace{\min_{i,l} \left(\frac{\alpha_{il}}{\beta_{il}} \right)}_{\mathcal{C}},$$

where the coefficient $\mathcal{C}$ is generally referred to as the SSP coefficient of the RK method (where $\frac{\alpha_{il}}{\beta_{il}} \equiv \infty$ for $\beta_{il} = 0$).

Much of the research in the area of SSP methods has focused on maximizing this coefficient for a given RK method of prescribed stage number and order; see, for example, [9,10,11,18,27,28,29]. These are typically referred to as optimal

Stages	SSPRK(s,2)+DG(2)		SSPRK(s,3)+DG(3)		SSPRK(s,4)+DG(4)	
s	μ	ν	μ	ν	μ	ν
2	0.3333	0.5000	...	...	...	...
3	0.5882	1.0000	0.2097	0.5000	...	...
4	0.7612	1.5000	0.3062	1.0000	...	...
5	0.8966	2.0000	0.4061	1.3253	0.2153	0.7541
6	1.0090	2.5000	0.4842	1.7592	0.2748	1.1500
7	1.1052	3.0000	0.5667	2.1440	0.3214	1.6605
8	1.1896	3.5000	0.6444	2.5536	0.3708	2.0730

Table 1 CFL restrictions for linear, μ, and TV, ν, stability of SSPRK(s,k)+DG(k) methods, using existing SSPRK time discretizations that have been optimized with respect to SSP coefficients. Note that μ is significantly less than ν in all cases.

SSP methods. Note, however, that for the case of RKDG methods the following condition must be satisfied in order to ensure both linear *and* TV-stability:

$$|c|\,\frac{\Delta t}{\Delta x}\;\leq\;\kappa=\min\big[\,\mu(p,\mathcal{S}),\;\nu(\mathcal{C})\,\big]. \tag{8}$$

Table 1 lists CFL restrictions for linear, μ, and TV, ν, stability of RKDG methods using existing s-stage, kth-order SSPRK methods, denoted SSPRK(s,k), paired with DG spatial discretizations of degree $p=k-1$, denoted DG($p+1$). (Note: such a method has an order of accuracy of k in the sense of local truncation errors; see [4].) It can be observed in Table 1 that μ is significantly less than ν in all cases, i.e., the condition on linear stability is the stricter condition. This motivates our interest in constructing SSPRK methods that are optimal with respect to linear stability as outlined in the next section.

Remark 1 We note that the definition of SSP methods can be extended to include negative β_{il}, in which case the RK method can still be written as a convex combination of forward Euler steps, provided a downwind operator, denoted $-\tilde{\mathbf{L}}$, is used in place of $\mathbf{L}$ for negative β_{il}. The downwind operator is simply the operator produced from a spatial discretization of (1) with $f(u)$ replaced by $-f(u)$. Here, we limit our investigation to SSP methods with nonnegative β_{il}. This particular subclass of explicit SSPRK methods is limited to orders $k\leq 4$. Furthermore, for $k=4$, the minimum number of stages required for the method to be SSP with nonnegative β_{il} is $s=5$; see, for example, Chapter 3 of [8].

3.1 Linear stability of Runge–Kutta methods

When applied to the prototypical scalar ODE $du/dt=\lambda u,\ \lambda\in\mathbb{C}$, a given s-stage, kth-order explicit RK method can always be expressed in the form

$$u^{n+1}=P_{s,k}(z)\,u^n,\quad z=\lambda\Delta t,$$

where $P_{s,k}(z)$ is the characteristic, or stability, polynomial of the RK method. For example, applying the classic three-stage, third-order SSPRK method to this equation written in Shu–Osher form yields

$$\begin{aligned}
u^{(1)} &= u^n + \Delta t \lambda u^n \\
u^{(2)} &= \frac{3}{4}u^n + \frac{1}{4}u^{(1)} + \frac{1}{4}\Delta t \lambda u^{(1)} \\
u^{n+1} &= \frac{1}{3}u^n + \frac{2}{3}u^{(2)} + \frac{2}{3}\Delta t \lambda u^{(2)}.
\end{aligned}$$

Substituting $u^{(1)}$ into the expression for $u^{(2)}$ and, in turn, $u^{(2)}$ into u^{n+1}, we obtain

$$u^{n+1} = \underbrace{\left(1 + z + \frac{1}{2}z^2 + \frac{1}{6}z^3\right)}_{P_{3,3}(z)} u^n,$$

where it can be observed that the stability polynomial in this case, $P_{3,3}(z)$, is the third-degree Taylor approximation of the exponential function. In order for the method to be linearly stable, we must have $|P_{s,k}| \leq 1$, where $|\cdot|$ is the complex modulus. The set in the complex plane for which this condition holds is the region of absolute stability $\mathcal{S}$ of the RK method, that is,

$$\mathcal{S} = \{z \in \mathbb{C} : \ |P_{s,k}| \leq 1\},$$

and in terms of $\mathcal{S}$, the condition for linear stability can be expressed as

$$\Delta t \lambda \subseteq \mathcal{S}; \tag{9}$$

in words, the coefficient λ, when scaled by the time step Δt, must be contained within the region of absolute stability.

In the case when the RK method is used to solve a system of ODEs, that is, when the scalar λ is replaced by a (normal) matrix $\mathbf{L}_h$, e.g., the semidiscrete DG equations, condition (9) must hold for each eigenvalue λ of $\mathbf{L}_h$ in order for the method to be linearly stable. (Note: if the matrix is non-normal, then it may be important to consider the pseudospectrum of $\mathbf{L}_h$; see, for example, [26].) Figure 2 shows, as an example, the stability domains for the SSPRK(2,2) and SSPRK(3,3) methods (left and right plots, respectively), along with the eigenvalues λ of the second- and third-degree DG spatial operators of the linear advection equation scaled by the maximum stable time steps (again, left and right plots, respectively). For nonlinear problems, it is common to study stability by imposing the spectral condition on the eigenvalues of linearizations of $f(u)$. Tests in the present work and in [25] suggest that the maximum stable CFL number for schemes that are optimized based on linear advection is also typically stable for nonlinear hyperbolic conservation laws.

One way to improve condition (9) is to design RK methods with larger regions of absolutely stability $\mathcal{S}$. This can be accomplished by using additional stages in the RK methods, which give rise to free parameters in the stability polynomials. These free parameters can then be adjusted to construct an $\mathcal{S}$ that accommodates the maximum scaled eigenvalue of the DG spatial operator. For example, a general five-stage, third-order RK method has a stability polynomial of the form

$$P_{5,3} = 1 + z + \frac{1}{2}z^2 + \frac{1}{6}z^3 + \gamma_4 z^4 + \gamma_5 z^5, \tag{10}$$

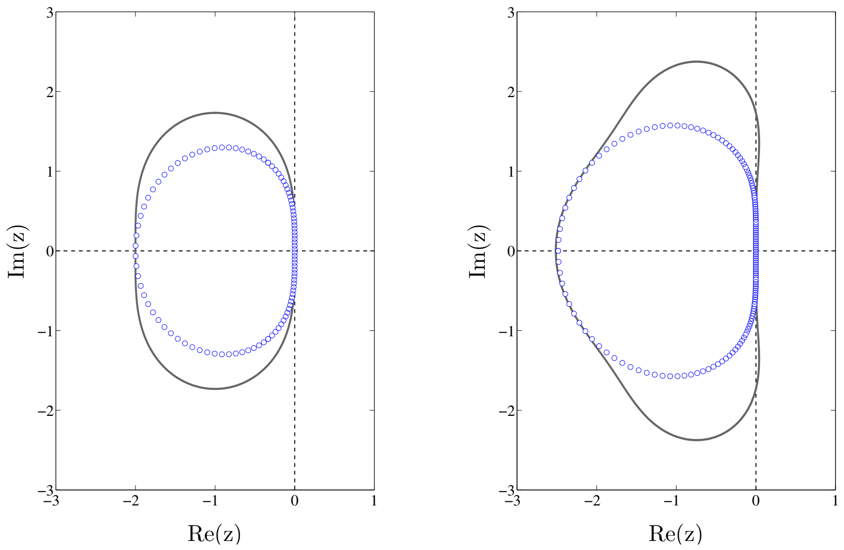

Fig. 2 Boundaries of the regions of absolute stability of the SSPRK(2,2) and SSPRK(3,3) methods (grey lines of the left and right plots, respectively) along with the the eigenvalues of the second- and third-order DG spatial operators for the linear advection case times the maximum stable time steps Δt (open blue circles of the left and right plots, respectively).

where γ_4 and γ_5 are free parameters that can be adjusted to maximize the CFL restrictions for RKDG linear stability.

It is important to note, of course, that an increase in the number of stages, while allowing for a larger time step Δt to be taken, also introduces additional computational effort per time step. Therefore, to make a fair comparison between methods, we define the *effective* time step of an s-stage RK method as

$$\Delta t_{\text{eff}} = \frac{\Delta t}{s}$$

and compare one method relative to another method of the same order by computing the relative change in the maximum stable effective time step. This is discussed in greater detail in Subsection 4.2.

4 Optimal SSPRK methods for DG

4.1 Method of construction

The construction of SSPRK methods (in canonical Shu-Osher form) that optimize the CFL restrictions for RKDG stability was carried out in the three main steps outlined below.

Step 1: First, optimal stability polynomials were found using an efficient procedure proposed by Ketcheson and Ahmadia [19], which is summarized in Algorithm 1.

Algorithm 1 Optimization of μ by bisection

```
$\Delta t_{\min} = 0$
$\Delta t_{\max} = \delta t > 0$
while $\Delta t_{\max} - \Delta t_{\min} \geq \epsilon$ do
    $\Delta t = (\Delta t_{\max} + \Delta t_{\min})/2$
    $r_{s,k} \leftarrow \underset{\gamma_{k+1},\gamma_{k+2},\ldots,\gamma_s}{\text{minimize}} \left[ \max_{\lambda \in \Lambda} \left( |P_{s,k}(\lambda \Delta t)| - 1 \right) \right]$
    if $r_{s,k} \leq 0$ then
        $\Delta t_{min} = \Delta t$
    else
        $\Delta t_{max} = \Delta t$
    end if
end while
return $\Delta t_\epsilon = \Delta t_{min}$ and $\{\gamma_i\}_{i=k+1}^{s}$
```

The first output of this algorithm satisfies

$$\lim_{\epsilon \to 0} |c| \frac{\Delta t_\epsilon}{\Delta x} = \mu_{\text{opt}}$$

under the condition that

$$|P_{s,k}(\lambda \Delta t_0)| = 1 \quad \Rightarrow \quad |P_{s,k}(\lambda \Delta t)| \leq 1 \text{ for all } 0 \leq \Delta t \leq \Delta t_0.$$

Although this condition does not necessarily hold for $k > 1$, in [19] Ketcheson and Ahmadia could find no situations with $s > k$ for which this condition is violated, and the algorithm is found to work well in practice. We note that we evaluate the eigenvalues of the DG spatial operator at a discrete number of points n, i.e., $\lambda \approx \lambda_n$ in Algorithm 1. The minimization problem of Algorithm 1 is solved using the `CVX` disciplined convex optimization toolbox for MATLAB [12,13]. The problem can be stated in just a few lines of code in the `CVX` environment and is generally solved in a matter of seconds.

Step 2: Next, we looked for a set of RK coefficients corresponding to the stability polynomial found in step 1 that maximizes the SSP coefficient $\mathcal{C}$. These RK coefficients must, of course, satisfy the usual conditions for order and consistency (see, e.g., [1]) *plus* the additional constraints placed on them from the values of the stability polynomial parameters $\{\gamma_i\}$ returned from Algorithm 1. At this stage of the process, it is useful to work with the Butcher form of the RK method, specifically,

$$\mathbf{u}^{(i)} = \mathbf{u}^n + \Delta t \sum_{l=1}^{s} a_{il} \mathbf{L}(\mathbf{u}^{(l)}), \quad i = 1, 2, ..., s$$
$$\mathbf{u}^{n+1} = \mathbf{u}^n + \Delta t \sum_{l=1}^{s} b_l \mathbf{L}(\mathbf{u}^{(l)}),$$

where we write the coefficients of the method, which are typically presented in a Butcher tableau, in the matrix form

$$\mathcal{A} = \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{b}^{\mathsf{T}} & 0 \end{bmatrix},$$

with $\mathbf{b} = [b_1, b_2, \dots, b_s]^\intercal$, $\mathbf{0}$ being a column vector of s zeros, and

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1s} \\ a_{21} & a_{22} & \dots & a_{2s} \\ \vdots & \vdots & \ddots & \vdots \\ a_{s1} & a_{s2} & \dots & a_{ss} \end{bmatrix},$$

which is lower triangular in the case of an explicit RK method.

The Butcher form is useful because it has simpler order condition formulations, a single well-defined SSP coefficient $\mathcal{C}$, and fewer decision variables for SSP optimization than the Shu–Osher form. We optimize the SSP coefficient using the following formulation of the problem:

$$\begin{aligned} \underset{\mathcal{A}}{\text{maximize}} \quad & \mathcal{C}(\mathcal{A}) \\ \text{subject to} \quad & \mathcal{A}(\mathbf{I} + \mathcal{C}\mathcal{A})^{-1} \geq 0 \\ & \|\mathcal{C}\mathcal{A}(\mathbf{I} + \mathcal{C}\mathcal{A})^{-1}\|_\infty \leq 1 \\ & \bar{\tau}_k(\mathcal{A}) = 0 \text{ (order conditions of order } k) \\ & \bar{v}_s(\mathcal{A}) = 0 \text{ (stability polynomial conditions)}, \end{aligned}$$

which is developed and presented in [8], where we have added the constraints on the set of coefficients $\{\gamma_i\}$ determined in step 1, denoted by $\bar{v}_s$. The optimization problem of this step was solved using `fmincon` in the MATLAB optimization toolbox. Note that if $\nu(\mathcal{C}) \geq \mu_{\text{opt}}$ found in step 1, then condition (8) has been successfully optimized. We note that this was indeed the case for every method considered here.

Step 3: After obtaining a set of Butcher coefficients with the maximum possible SSP coefficient, we can easily transform it to the canonical Shu–Osher form (the Shu–Osher form that is a convex combination of forward Euler steps all of equal size) by first computing the following matrices and vector:

$$\begin{aligned} \widehat{\boldsymbol{\alpha}} &= \mathcal{C}\mathcal{A}\left(\mathbf{I} + \mathcal{C}\mathcal{A}\right)^{-1} \\ \widehat{\boldsymbol{\beta}} &= \mathcal{A}\left(\mathbf{I} + \mathcal{C}\mathcal{A}\right)^{-1} \\ \widehat{\boldsymbol{\gamma}} &= \left(\mathbf{I} + \mathcal{C}\mathcal{A}\right)^{-1} \mathbf{e}, \end{aligned}$$

where $\mathbf{e} = [1, 1, \dots, 1]^\intercal$, and then computing the α_{il} and β_{il} of (7) for $l = 0, 1, \dots, s-1$ via

$$\alpha_{il} = \begin{cases} \widehat{\alpha}_{i+1,l+1} + \widehat{\gamma}_{i+1} & l = 0 \\ \widehat{\alpha}_{i+1,l+1} & \text{otherwise} \end{cases}, \qquad \beta_{il} = \widehat{\beta}_{i+1,l+1}, \quad i = 1, 2, \dots, s;$$

see, for example, [8] and [15] for details and a derivation of the above formulas.

4.2 Results

Using the three-step procedure outlined above, new SSPRK methods with up to eight stages were derived for orders $k = 2$, 3, and 4. Table 2 summarizes the results, where we indicate the conditions for linear, μ, and TV-stability, ν, for each method as well as the percent improvement in the CFL restrictions; see condition

Stages	SSPRK(s,2)+DG(2)			SSPRK(s,3)+DG(3)			SSPRK(s,4)+DG(4)		
s	μ	ν	ε(%)	μ	ν	ε(%)	μ	ν	ε(%)
2	0.3333	0.5000	...	...	...	...	...	...	...
3	0.5904	0.9470	**0.37**	0.2097	0.5000	...	...	...	...
4	0.8257	1.2298	**8.47**	0.3160	0.8417	**3.27**	...	...	...
5	1.0520	1.5392	**17.32**	0.4330	1.1937	**6.62**	0.2201	0.8528	**2.19**
6	1.2740	1.8425	**26.26**	0.5510	1.5355	**13.80**	0.2861	1.1139	**9.71**
7	1.4935	2.1479	**32.13**	0.6686	1.8704	**17.98**	0.3527	1.1651	**14.56**
8	1.7114	2.4532	**43.86**	0.7852	2.1976	**21.85**	0.4213	1.7711	**18.24**

Table 2 CFL restrictions for linear, μ, and TV, ν, stability of SSPRK(s,k)+DG(k) methods, using the new SSPRK time discretizations that have been optimized with respect to linear stability for DG. The percent improvements in CFL restrictions, denoted ε, are indicated in bold.

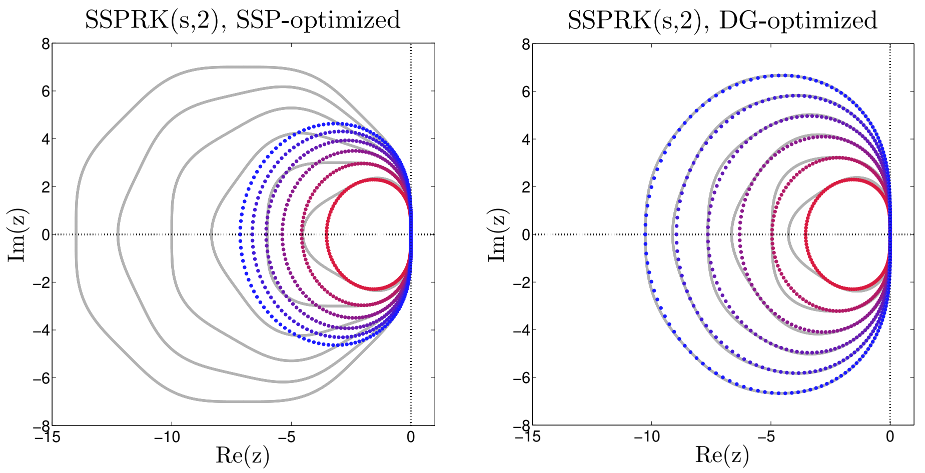

Fig. 3 Plots of the stability domains (grey) of the SSP-optimized (left) and DG-optimized (right) SSPRK methods with up to eight stages, along with $z = \Delta t\lambda$ (dots), where the discrete eigenvalues λ_n of the second-order DG spatial operator are multiplied by the maximum stable time step Δt for $s = 2$ to $s = 8$ (red$\rightarrow$blue as $s \rightarrow 8$).

(8), compared to the existing s-stage SSPRK methods of the same order; cf. Table 1. The α and β coefficients of the new methods, along with their SSP coefficients, are included in Appendix A in Tables 7–21. Figures 3–5 show the stability domains and maximum $\Delta t\lambda$ values of the RKDG methods using existing SSP-optimized RK methods and the new DG-optimized RK methods. The figures show that the new methods have stability domains that are better suited to the shape of $\Delta t\lambda$ in the complex plane, allowing for a larger Δt.

As noted above in Section 3, when considering the overall efficiency of the methods, it is important to compare their effective time steps. For a given order k, the method that requires the least amount of computational effort *per time step* is, of course, the one with the fewest number of stages, denoted $\hat{s}_k$, that is required for the method to be SSP considering nonnegative β. (Note: $\hat{s}_k = k$ for $k = 2$ and $k = 3$, while $\hat{s}_k = k+1$ for $k = 4$ as mentioned above.) These $\hat{s}_k$-stage methods are used as the basis of comparison for all other methods of order k. Specifically, we compute the percent change in effective time step of a given SSPRK(s,k) method

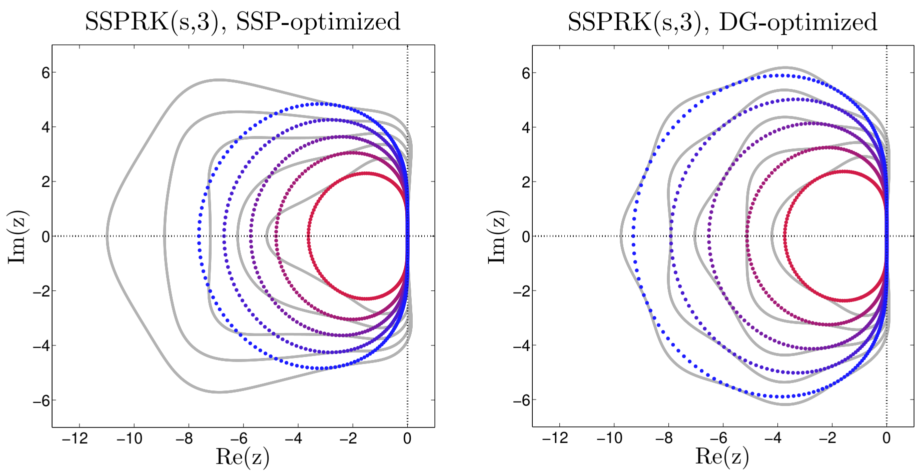

Fig. 4 Plots of the stability domains (grey) of the SSP-optimized (left) and DG-optimized (right) SSPRK methods with up to eight stages, along with $z = \Delta t \lambda$ (dots), where the discrete eigenvalues λ_n of the third-order DG spatial operator are multiplied by the maximum stable time step Δt for $s = 3$ to $s = 8$ (red$\rightarrow$blue as $s \rightarrow 8$).

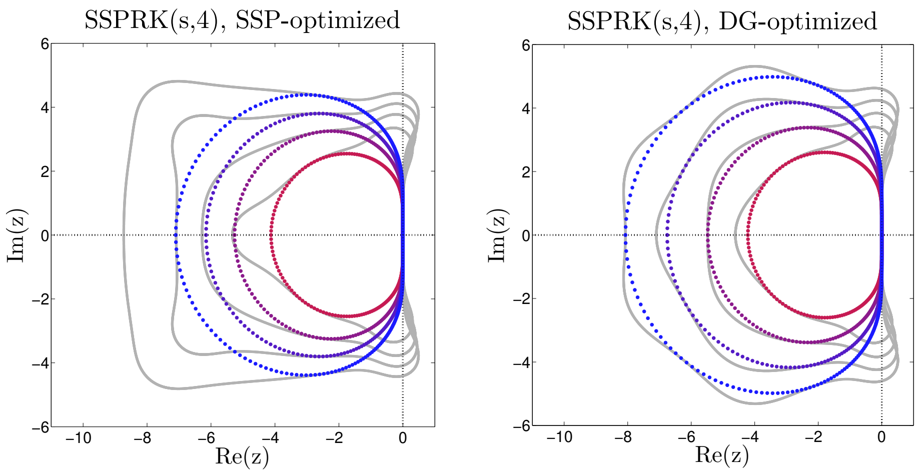

Fig. 5 Plots of the stability domains (grey) of the SSP-optimized (left) and DG-optimized (right) SSPRK methods with up to eight stages, along with $z = \Delta t \lambda$ (dots), where the discrete eigenvalues λ_n of the fourth-order DG spatial operator are multiplied by the maximum stable time step Δt for $s = 5$ to $s = 8$ (red$\rightarrow$blue as $s \rightarrow 8$).

relative to the existing SSPRK($\hat{s}_k$,k) method. We refer to these values as relative efficiencies (denoted by $\Delta(s,k)\%$ in Table 3), which are reported, along with the corresponding CFL restrictions for the methods, in Table 3 for both the existing SSP-optimized methods and the new, DG-optimized methods constructed here.

With regard to Table 3, it is interesting to observe that, while the existing SSPRK methods peak in relative efficiency after the addition of only a few additional stages, the relative efficiencies of the new methods continue to increase with

Existing SSPRK methods

Stages, s	SSPRK(s,2)+DG(2)		SSPRK(s,3)+DG(3)		SSPRK(s,4)+DG(4)	
	CFL	$\Delta(s,2)$%	CFL	$\Delta(s,3)$%	CFL	$\Delta(s,4)$%
2	0.3333	...	...	...	...	...
3	0.5882	17.65	0.2097	...	...	...
4	0.7612	14.19	0.3062	9.51	...	...
5	0.8966	7.60	0.4061	16.19	0.2153	...
6	1.0090	0.91	0.4842	15.45	0.2748	6.36
7	1.1052	−5.26	0.5667	15.81	0.3214	6.63
8	1.1896	−10.77	0.6444	15.23	0.3708	7.64

New SSPRK methods

Stages, s	SSPRK(s,2)+DG(2)		SSPRK(s,3)+DG(3)		SSPRK(s,4)+DG(4)	
	CFL	$\Delta(s,2)$ %	CFL	$\Delta(s,3)$ %	CFL	$\Delta(s,4)$ %
2	0.3333	...	...	...	...	...
3	0.5904	18.09	0.2097	...	...	...
4	0.8257	23.87	0.3160	13.02	...	...
5	1.0519	26.24	0.4330	23.89	0.2201	2.23
6	1.2740	27.41	0.5510	31.38	0.2861	10.74
7	1.4935	28.03	0.6686	36.64	0.3527	17.01
8	1.7114	28.37	0.7852	40.42	0.4213	22.30

Table 3 CFL restrictions and the percentage change in effective time step of SSPRK(s,k)+DG(k) methods using existing SSPRK time discretizations that have been optimized with respect to SSP coefficients (top) and the new SSPRK time discretizations that have been optimized with respect to the DG spatial operators (bottom). Percent changes in effective time steps, $\Delta(s,k)$, are computed relative to the existing SSPRK method that requires the minimum number of stages for a given order k.

the addition of each new stage. For example, the existing second-order SSPRK methods peak in relative efficiency at $s = 5$ at a value of 17.65%, while the new second-order methods continue to increase with a relative efficiency of 28.37% for $s = 8$. (Note: beyond $s = 8$ we found only marginal gains in relative efficiency; for example, a DG-optimzed $s = 9$, second-order method resulted in an efficiency increase of less than 0.20% over the $s = 8$ method.) The most significant gain in relative efficiency was realized in the third-order case, where the new eight-stage, third-order method offers a relative efficiency gain of 40% over the classic three-stage, third-order method. Finally, we note that we have not considered the effect of storage with respect to relative efficiency here, which may be an important factor for "large" problem sizes.

Finally, to highlight the differences between the work presented here and another effort in this area, we conclude this section by summarizing the recent work of Toulorge and Desmet [31], where three specific RK methods were derived for DG spatial discretizations under two different optimization scenarios — what the authors refer to as the "free element size" (RKF) and "constrained element size" (RKC) scenarios. Under the first scenario, RK stability polynomials of order $k = 3$ and 4 with $s = k + 1$ to $s = k + 4$ stages were first sought that minimized the (mean) computational cost required to achieve a prescribed error tolerance over a range of DG spatial discretizations (polynomial degrees of $p = 1$ to 10 were considered). From these results, the stability polynomial that was the most efficient over the full range of DG spatial discretizations that were considered was then selected to be used in the development of a corresponding RK method, what is referred to as the RKF84 method by the authors, i.e., an eight-stage, fourth-order RK method optimized under the RKF scenario. For the RKC scenario, the authors considered the same set of RK stability polynomials and range of DG spatial discretizations, this time optimizing with respect to (mean) linear stability conditions. Of these stability polynomials, two were selected to be used in the development of corresponding RK methods, which resulted in their RKC73 and RKC84 methods ($\mu = 0.4132$ and $\mu = 0.6406$, respectively, when used with DG spatial discretizations of matching order, cf. Table 2). SSP conditions were not considered in the derivation of the three RK methods, with the coefficients given in Williamson low-storage form [34]. We note that the CFL restrictions obtained for these methods were stricter than the CFL restrictions of the SSPRK(7,3) and SSPRK(8,4) methods derived here when used in conjunction with a DG spatial discretization of the same order.

5 Numerical test cases

In this section, we present the results of two numerical test cases that demonstrate the performance of the new methods. First, we verify the stability of the new SSPRK DG methods for a simple linear problem, where we check the CFL restrictions obtained in the previous section and evaluate the computational savings achieved in practice using the new methods. Within the context of this linear problem, we also demonstrate the lack of higher order convergence that occurs if the linear CFL restriction is violated, even though the solution remains TV-stable by respecting the TV CFL restriction. Second, we apply the new methods to a nonlinear problem, where we demonstrate convergence in the nonlinear case and evaluate the performance of the methods in the presence of discontinuities.

5.1 Test Case 1: Linear advection of a sine wave

The RKDG methods were first applied to solve the one-dimensional, linear advection equation, i.e.,

$$\frac{\partial x}{\partial t} + \frac{\partial}{\partial x}(cu) = 0, \quad \text{for } x \in [-\pi, \pi], \quad t \in (0, T],$$

with periodic boundary conditions and an initial condition $u_0(x) = \sin(2\pi x/D)$, where D is the length of the domain, i.e., $D = 2\pi$. In our numerical experiments, we take $c = 1$ and run to a final time of $T = 315$, which allows the sine wave to cross the domain approximately 50 times.

Table 4 presents theoretical and numerical CFL restrictions and relative efficiencies of the new RKDG methods using a mesh of $N = 50$ elements of equal width. Numerical CFLs for a given method were obtained by starting with the theoretical CFL restriction, μ_{opt}, determined from step 1 of the optimization procedure. If the run was stable at μ_{opt} (which was always the case), then the CFL was increased by 0.0001 until instability was observed in the solution. The CFL restriction of the last stable run was then recorded as the numerical CFL restriction. From Table 4, very good agreement can be observed between the theoretical and

SSPRK(s,2)+DG(2) methods

Stages, s	μ		$\Delta(s,2)\%$	
	Theor.	Num.	Theor.	Num.
2	0.3333	0.3340	...	...
3	0.5904	0.5917	18.09	18.74
4	0.8257	0.8274	23.87	24.16
5	1.0519	1.0540	26.24	26.60
6	1.2740	1.2762	27.41	27.70
7	1.4935	1.4962	28.03	28.19
8	1.7114	1.7137	28.37	28.49

SSPRK(s,3)+DG(3) methods

Stages, s	μ		$\Delta(s,3)\%$	
	Theor.	Num.	Theor.	Num.
3	0.2097	0.2099	...	...
4	0.3160	0.3164	13.02	13.00
5	0.4330	0.4334	23.89	24.14
6	0.5510	0.5515	31.38	31.23
7	0.6686	0.6692	36.64	36.21
8	0.7852	0.7860	40.42	39.70

SSPRK(s,4)+DG(4) methods

Stages, s	μ		$\Delta(s,4)\%$	
	Theor.	Num.	Theor.	Num.
5	0.2201	0.2202	2.23	2.25
6	0.2861	0.2861	10.74	10.56
7	0.3527	0.3528	17.01	16.52
8	0.4213	0.4214	22.30	21.80

Table 4 Comparison of the theoretical and numerical CFL restrictions and relative efficiencies for the second-, third-, and fourth-order RKDG methods (top, middle, and bottom, respectively) using the new DG-optimized SSPRK methods.

the numerically obtained CFLs, with the largest discrepancy being only 0.22%. The relative efficiencies of the methods were tested by comparing computational run times using the maximum allowable time step as dictated by the theoretical CFL restriction, i.e., $\Delta t = \mu_{\text{opt}} \Delta x$. Once again, good agreement can be observed between predicted and obtained values.

Although not shown here, all methods displayed optimal convergence rates in L^2 with respect to mesh refinement when the linear stability constraints were respected. To demonstrate the degradation of high-order convergence when the linear stability requirements are not met, we present results from running the test case twice using the new SSPRK(3,2) method — once with a time step determined by the requirement for linear stability μ, and once using a time step determined by the less strict requirement for TV-stability ν. In order to ensure TV-stability, the generalized slope limiter of [4] was applied in both cases, using a value of the slope limiter parameter M as suggested in that work. Table 5 shows the convergence in the L^2 error using both conditions. Optimal second-order convergence is observed when the linear stability condition is met; however, convergence degrades to first-order when the condition for linear stability is not enforced, even though the solution remains TV-stable. This example emphasizes the point that it is the linear

Δx	$\Delta t/\Delta x = 0.5904 < \mu$		$\mu < \Delta t/\Delta x = 1.2000 < \nu$	
	Error	Order	Error	Order
$2\pi/50$	1.54E-02	...	4.76E-01	...
$2\pi/100$	3.86E-03	2.00	2.39E-01	0.99
$2\pi/200$	9.65E-04	2.00	1.16E-01	1.05
$2\pi/400$	2.41E-04	2.00	5.78E-02	1.00

Table 5 Convergence of the second-order RKDG method using the new SSPRK(3,2) method, which displays optimal second-order convergence when $\Delta t/\Delta x < \mu$ but which degenerates to first-order when $\mu < \Delta t/\Delta x < \nu$.

stability requirement that must be respected in practice in order to maintain both stability and high-order convergence.

5.2 Test Case 2: Burgers' equation

For our second test case, we apply the new RK methods to Burgers' equation

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}\left(\frac{1}{2}u^2\right) = 0, \quad \text{for } x \in [0, 200], \quad t \in (0, T],$$

with periodic boundary conditions, and the sine wave initial condition used in the previous problem with $D = 200$. The exact solution to the problem forms a shock at time $t = 100/\pi$, making it a good test for the nonlinear stability properties of the methods.

First, in order to demonstrate proper error convergence in L^2 with respect to mesh refinement, the problem was run to time $T = 22$, well before the formation of the shock, using the maximum allowable time step dictated by the CFL requirement for linear stability, and applying the slope limiter as mentioned above. Beginning with a 100 element mesh ($\Delta x = 2.00$), and doubling the resolution to 200, 400, and 800 elements ($\Delta x = 1.00, 0.50, 0.25$), the L^2 error in the final solution was calculated. All methods showed the expected order of convergence under mesh refinement. A selection of methods with their computed rates of convergence is presented in Table 6. Note that similar errors are observed between schemes of the same order.

Next, to test the TV-stability of the new DG-optimized RKDG methods, the numerical solution was computed at time $T = 32$, just after the formation of the shock. The goal is to avoid the formation of spurious oscillations that can appear around steep fronts in numerical solutions. The problem was run on a 40 element spatial mesh ($\Delta x = 5$) with a time step dictated by (8). The TVDM slope limiter of Cockburn and Shu was once again applied, with an appropriate choice for the parameter M. Figure 6 shows plots of the solutions obtained using the new SSPRK(3,2), SSPRK(4,3), and SSPRK(5,4) methods. In each case, the TV of the numerical solution was verified numerically to be nonincreasing in the means. Similar behavior around the front was observed for all of the DG-optimized RKDG methods.

SSPRK(s,2)+DG(2) methods

Δx	SSPRK(3,2)+DG(2)		SSPRK(8,2)+DG(2)	
	Error	Order	Error	Order
2.000	7.72E-03	...	9.95E-03	...
1.000	1.87E-03	2.04	2.01E-03	2.31
0.500	4.62E-04	2.02	4.58E-04	2.13
0.250	1.15E-04	2.01	1.10E-04	2.06

SSPRK(s,3)+DG(3) methods

Δx	SSPRK(4,3)+DG(3)		SSPRK(8,3)+DG(3)	
	Error	Order	Error	Order
2.000	2.00E-04	...	3.20E-04	...
1.000	2.50E-05	3.00	3.39E-05	3.24
0.500	3.21E-06	2.96	4.03E-06	3.07
0.250	4.14E-07	2.96	5.01E-07	3.01

SSPRK(s,4)+DG(4) methods

Δx	SSPRK(5,4)+DG(4)		SSPRK(8,4)+DG(4)	
	Error	Order	Error	Order
2.000	9.82E-06	...	1.40E-05	...
1.000	6.10E-07	4.01	7.86E-07	4.15
0.500	3.89E-08	3.97	4.70E-08	4.06
0.250	2.58E-09	3.92	2.89E-09	4.02

Table 6 Convergence of the SSPRK($k+1$,k)+DG(k) and SSPRK(8,k)+DG(k) methods in the L^2 norm using the new SSPRK methods of order k = 2, 3, and 4.

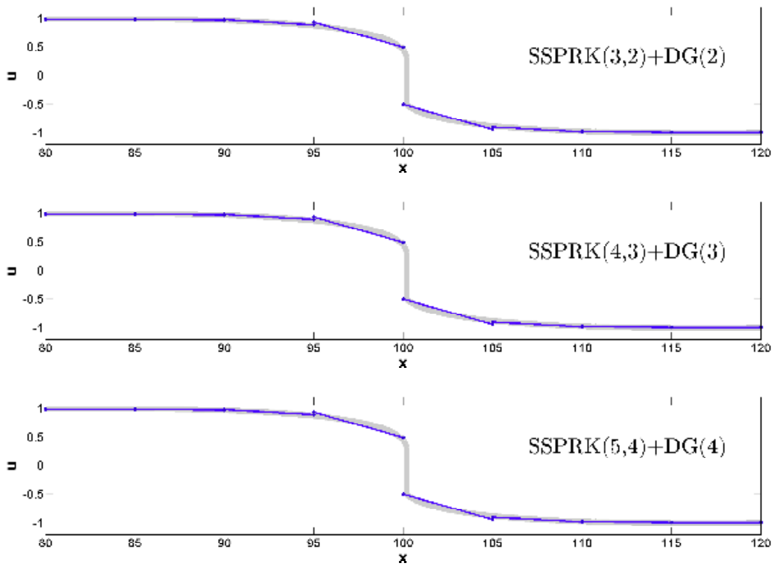

Fig. 6 Comparison of the RKDG solutions (blue lines) to the exact solution for test case 2 (grey line in each plot) at time $t = 32$ in the vicinity of the shock.

6 Conclusions

New SSPRK methods have been derived that optimize allowable time step size while maintaining stability and high-order convergence when applied to DG spatial discretizations. These new methods represent the best available RKDG methods in terms of computational efficiency, with significant improvements over schemes using SSPRK methods optimized with respect to SSP coefficients. In direct comparisons of allowable time step sizes between DG-optimized and SSP-optimized methods of similar order and number of stages, improvements of up to 43.86% were acheived. In terms of overall computational savings of the new methods relative to minimal-stage methods of a given order, improvements in effective time step size of up to 40% were realized with the new DG-optimized methods. In contrast, the maximum savings of the existing SSP-optimized methods over the minimal stage method is only around 18%. The linear and nonlinear stability properties were verified by applying the RKDG methods to both linear and nonlinear test cases. The new stability requirements and efficiency improvements were verified numerically, and the methods were shown to have high-order error convergence under mesh refinement. We note that the methods in this paper have been optimized specifically for one spatial dimension; however, a similar optimization approach can be applied in multi-dimensions; see [19]. We note that preliminary results from the optimization of 2D RKDG methods on equilateral triangular meshes show only modest improvements (at best $\sim 5\%$) in the timestep restriction over the new optimal 1D RKDG methods presented here.

Future work in this area includes investigating multirate RK schemes that use the new DG-optimized RK time steppers as "base methods"; see, for example, [6, 7], and the development of optimal multistep and multistep-multistage time steppers for DG spatial discretizations, which may offer additional computational savings and allow for the extension of the method to orders greater than four. Finally, the work in this paper can also be extended to develop low-storage, minimal error SSPRK methods with optimal stability constraints.

Acknowledgements The first and second author acknowledge support by National Science Foundation grants DMS-0915118 and DMS-1217218.

References

1. Butcher, J.C.: Numerical Methods for Ordinary Differential Equations, 2nd edn. John Wiley & Sons, Hoboken, NJ (2008)
2. Cheng, Y., Li, F., Qiu, J., Xu, L.: Positivity-preserving DG and central DG methods for ideal MHD equations. Journal of Computational Physics **238**, 255–280 (2013)
3. Cockburn, B., Shu, C.W.: The Runge–Kutta local projection P^1-discontinuous Galerkin finite element method for scalar conservation laws. Mathematical Modelling and Numerical Analysis **25**(3), 337–361 (1989)
4. Cockburn, B., Shu, C.W.: TVB Runge–Kutta local projection discontinuous Galerkin finite element method for scalar conservation laws II: general framework. Mathematics of Computation **52**, 411–435 (1989)
5. Cockburn, B., Shu, C.W.: Runge–Kutta discontinuous Galerkin methods for convection-dominated problems. Journal of Scientific Computing **16**(3), 173–261 (2001)
6. Constantinescu, E.M.: Multirate timestepping methods. Journal of Scientific Computing **33**, 239–278 (2007)

7. Dawson, C., Trahan, C.J., Kubatko, E.J., Westerink, J.J.: A parallel local timestepping Runge–Kutta discontinuous Galerkin method with applications to coastal ocean modeling. Computer Methods in Applied Mechanics and Engineering **259**(1), 154–165 (2013)
8. Gottlieb, S., Ketcheson, D., Shu, C.W.: Strong Stability Preserving Runge–Kutta and Multistep Time Discretizations. World Scientific (2011)
9. Gottlieb, S., Shu, C.W.: Total variation diminishing Runge-Kutta schemes. Mathematics of Computation **67**, 73–85 (1998)
10. Gottlieb, S., Shu, C.W.: Strong stability preserving properties of Runge–Kutta time discretization methods for linear constant coefficient operators. Journal of Scientific Computing **18**(1), 83–109 (2003)
11. Gottlieb, S., Shu, C.W., Tadmor, E.: Strong stability-preserving high-order time discretization methods. SIAM Review **43**(1), 89–112 (2001)
12. Grant, M., Boyd, S.: Graph implementations for nonsmooth convex programs. In: V. Blondel, S. Boyd, H. Kimura (eds.) Recent Advances in Learning and Control, Lecture Notes in Control and Information Sciences, pp. 95–110. Springer-Verlag Limited (2008). `http://stanford.edu/~boyd/graph_dcp.html`
13. Grant, M., Boyd, S.: CVX: Matlab software for disciplined convex programming, version 2.0 beta. `http://cvxr.com/cvx` (2012)
14. Harten, A.: High resolution schemes for hyperbolic conservation laws. Journal of Computational Physics **49**, 357–393 (1983)
15. Higueras, I.: Representations of Runge–Kutta methods and strong stability preserving methods. SIAM Journal of Numerical Analysis **43**(3), 924–948 (2005)
16. van der Houwen, P.: Explicit Runge-Kutta formulas with increased stability boundaries. Numerische Mathematik **20**, 149–164 (1972)
17. Jeltsch, R., Nevanlinna, O.: Largest disk of stability of explicit Runge-Kutta methods. BIT Numerical Mathematics **18**, 500–502 (1978)
18. Ketcheson, D.I.: Highly efficient strong stability preserving Runge–Kutta methods with low-storage implementations. SIAM Journal on Scientific Computing **30**(4), 2113–2136 (2008)
19. Ketcheson, D.I., Ahmadia, A.: Optimal stability polynomials for numerical integration of initial value problems. Communications in Applied Mathematics and Computational Science **7**(2), 247–271 (2013)
20. Kinnmark, I.P., Gray, W.G.: One step integration methods with maximum stability regions. Mathematics and Computers in Simulation **26**, 87–92 (1984)
21. Kubatko, E.J., Bunya, S., Dawson, C., Westerink, J.J.: Dynamic p-adaptive Runge–Kutta discontinuous Galerkin methods for the shallow water equations. Computer Methods in Applied Mechanics and Engineering **198**, 1766–1774 (2009)
22. Kubatko, E.J., Westerink, J.J., Dawson, C.: Semidiscrete discontinuous Galerkin methods and stage exceeding order strong stability preserving Runge-Kutta time discretizations. Journal of Computational Physics **222**, 832–848 (2007)
23. Mirabito, C., Dawson, C., Kubatko, E.J., Westerink, J.J., Bunya, S.: Implementation of a discontinuous Galerkin morphological model on two-dimensional unstructured meshes. Computer Methods in Applied Mechanics and Engineering **200**, 189–207 (2011)
24. Parsani, M., Ketcheson, D.I.: Design of optimal explicit linearly stable strong stability preserving Runge-Kutta schemes for the spectral difference method. In: International Conference on Spectral and High Order Methods (2012)
25. Parsani, M., Ketcheson, D.I., Deconinck, W.: Optimized explicit Runge-Kutta schemes for the spectral difference method applied to wave propagation problems. SIAM Journal on Scientific Computing **35**(2), A957–A986 (2013)
26. Reddy, S.C., Trefethen, L.N.: Stability of the method of lines. Numerische Mathematik pp. 235–267 (1992)
27. Ruuth, S.J.: Global optimization of explicit strong-stability-preserving Runge–Kutta methods. Mathematics of Computation **75**, 183–207 (2006)
28. Spiteri, R.J., Ruuth, S.J.: A new class of optimal high-order strong-stability-preserving time discretization methods. SIAM Journal on Numerical Analysis **40**, 469–491 (2002)
29. Spiteri, R.J., Ruuth, S.J.: Non-linear evolution using optimal fourth-order strong-stability-preserving Runge–Kutta methods. Mathematics and Computers in Simulation **62**, 125–135 (2003)
30. Sun, T., Qiu, J.: LWDG method for a multi-class traffic flow model on an inhomogeneous highway. Advances in Applied Mathematics and Mechanics **1**(3), 438–450 (2009)

31. Toulorge, T., Desmet, W.: Optimal Runge–Kutta schemes for discontinuous Galerkin space discretizations applied to wave propagation problems. Journal of Computational Physics **231**(4), 2067–2091 (2012)
32. Trahan, C.J., Dawson, C.: Local time-stepping in Runge–Kutta discontinuous Galerkin finite element methods applied to the shallow-water equations. Computer Methods in Applied Mechanics and Engineering **217**, 139–152 (2012)
33. Vichnevetsky, R.: New stability theorems concerning one-step numerical methods for ordinary differential equations. Mathematics and Computers in Simulation **25**, 199–205 (1983)
34. Williamson, J.H.: Low-storage Runge–Kutta schemes. Journal of Computational Physics **35**, 48–56 (1980)
35. Xing, Y., Shu, C.W.: High order well-balanced finite volume WENO schemes and discontinuous Galerkin methods for a class of hyperbolic systems with source terms. Journal of Computational Physics **214**, 567–598 (2010)
36. Xing, Y., Zhang, X., Shu, C.W.: Positivity-preserving high order well-balanced discontinuous Galerkin methods for the shallow water equations. Advances in Water Resources **33**, 1476–1493 (2010)

A Shu–Osher form coefficients of the new Runge–Kutta methods

Table 7 SSPRK(3,2)

α_{il}		$\mathcal{C}$ = 1.893921369918281
1.000000000000000	0	0
0.087353119859156	0.912646880140844	0
0.344956917166841	0	0.655043082833159

β_{il}		
0.528005024856522	0	0
0	0.481882138633993	0
0.022826837460491	0	0.345866039233415

Table 8 SSPRK(4,2)

α_{il}			$\mathcal{C}$ = 2.459513555939448
1.000000000000000	0	0	0
0.394806441339829	0.605193558660171	0	0
0.002797307087390	0	0.997202692912610	0
0.252860909354373	0	0	0.747139090645627

β_{il}			
0.406584463657504	0	0	0
0	0.246062298456822	0	0
0.013637216641451	0	0.405447122055692	0
0.016453567333598	0	0	0.303775146447707

Table 9 SSPRK(5,2)

α_{il}				$\mathcal{C}$ = 3.078432757856577
1.000000000000000	0	0	0	0
0.235593265061659	0.764406734938341	0	0	0
0.174017972351526	0	0.825982027648475	0	0
0.235264368870758	0.000058643383967	0	0.764676987745275	0
0.141720372339803	0.095374613155521	0.000311763705780	0	0.762593250798895

β_{il}				
0.324840618151514	0	0	0	0
0	0.248310356296551	0	0	0
0.108822380501601	0	0.268312512443371	0	0
0.054392262422093	0.000019049753098	0	0.248398145385413	0
0.000000180291569	0.030981548293401	0.000101273514903	0	0.247721262987686

Table 10 SSPRK(6,2)

α_{il} $\mathcal{C} = 3.685003559472798$

1.000000000000000	0	0	0	0	0
0.176902819560407	0.823097180439593	0	0	0	0
0.015893151207488	0	0.984106848792512	0	0	0
0.153504267159468	0.000003730459625	0	0.846492002380908	0	0
0.180356799710441	0.227796438692973	0.000004416728347	0	0.591842344868240	0
0.098962308653140	0.000000000411893	0.151738038514171	0.019744621964792	0	0.729555030456003

β_{il}

0.271370158498047	0	0	0	0	0
0	0.223364012315188	0	0	0	0
0.174831877653527	0	0.267057231535837	0	0	0
0.024088031667553	0.000001012335420	0	0.229712668853436	0	0
0.042683362900909	0.061817155673403	0.000001198568272	0	0.160608350932750	0
0.000000000265385	0.000000000111776	0.041177175561773	0.005358101192070	0	0.197979464247893

Table 11 SSPRK(7,2)

α_{il}						$\mathcal{C} = 4.295752077809973$
1.000000000000000	0	0	0	0	0	0
0.000058693366670	0.999941306633330	0	0	0	0	0
0.064464637955751	0	0.935535362044249	0	0	0	0
0.220086300845449	0.102225195325932	0	0.677688503828619	0	0	0
0.109581884408778	0.071060761259314	0.005348397939225	0	0.814008956392683	0	0
0.155551745312158	0.125802445334275	0.078456975942849	0.018610166400853	0	0.621578667009864	0
0.068396686509772	0.054238497539825	0.022030341614959	0.004628734371986	0.103696237861290	0	0.747009502102168

β_{il}						
0.232788108318814	0	0	0	0	0	0
0	0.232774445201016	0	0	0	0	0
0.165502968997453	0	0.217781507195638	0	0	0	0
0.103058857428462	0.023796809842445	0	0.157757824835672	0	0	0
0.030644252256650	0.016542100189251	0.001245043438808	0	0.189491605113225	0	0
0.018654279847214	0.029285313271247	0.018263851014151	0.004332225431953	0	0.144696122064557	0
0.000000001040139	0.012626077240350	0.005128401550164	0.001077514318365	0.024139251051507	0	0.173894928890543

Table 12 SSPRK(8,2)

α_{il}							$\mathcal{C} = 4.906377753898920$
1.000000000000000	0	0	0	0	0	0	0
0.000022706789062	0.999977293210938	0	0	0	0	0	0
0.040163352005589	0	0.959836647994411	0	0	0	0	0
0.068277481460787	0.000794112060878	0	0.930928406478335	0	0	0	0
0.203201797047416	0.049250734246472	0.028073821192189	0	0.719473647513923	0	0	0
0.135232895616102	0.000233842652217	0.057212625463321	0.005935142172269	0	0.801385494096090	0	0
0.330663126197853	0	0	0.036269305920717	0.087351183650948	0	0.545716384230481	0
0.072377654137843	0.001047652272129	0	0.005351824870836	0.206003319412558	0.025024389094107	0	0.690195160212526

β_{il}							
0.203816348874755	0	0	0	0	0	0	0
0	0.203811720859914	0	0	0	0	0	0
0.203690674361979	0	0.195630401110404	0	0	0	0	0
0.036121835699129	0.000161853020846	0	0.189738428872208	0	0	0	0
0.084801239646524	0.010038104833517	0.005721903734355	0	0.146640491947890	0	0	0
0	0.000047660955586	0.011660868431473	0.001209679007605	0	0.163335465447857	0	0
0.058391312304979	0	0	0.007392277508982	0.017803599321625	0	0.111225920954990	0
0.004977697629959	0.000213528660996	0	0.001090789404991	0.041986844418748	0.005100379617982	0	0.140673057565544

Table 13 SSPRK(4,3)

α_{il}			$\mathcal{C} = 1.683339717642499$
1.000000000000000	0	0	0
0.522361915162541	0.477638084837459	0	0
0.368530939472566	0	0.631469060527434	0
0.334082932462285	0.006966183666289	0	0.658950883871426

β_{il}			
0.594057152884440	0	0	0
0	0.283744320787718	0	0
0.000000038023030	0	0.375128712231540	0
0.116941419604231	0.004138311235266	0	0.391454485963345

Table 14 SSPRK(5,3)

α_{il}				$\mathcal{C} = 2.387300839230550$
1.000000000000000	0	0	0	0
0.495124140877703	0.504875859122297	0	0	0
0.105701991897526	0	0.894298008102474	0	0
0.411551205755676	0.011170516177380	0	0.577278278066944	0
0.186911123548222	0.013354480555382	0.012758264566319	0	0.786976131330077

β_{il}				
0.418883109982196	0	0	0	0
0	0.211483970024081	0	0	0
0.000000000612488	0	0.374606330884848	0	0
0.046744815663888	0.004679140556487	0	0.241812120441849	0
0.071938257223857	0.005593966347235	0.005344221539515	0	0.329651009373300

Table 15 SSPRK(6,3)

α_{il}					$\mathcal{C} = 3.071058071923395$
1.000000000000000	0	0	0	0	0
0.271376652410776	0.728623347589224	0	0	0	0
0.003607665467954	0	0.996392334532046	0	0	0
0.295174024904477	0.104490494022953	0	0.600335481072570	0	0
0.300088895805571	0.000000004174982	0.000038417983374	0	0.699872682036073	0
0.057902281374384	0.003951957060919	0.179481122980769	0.126656280556504	0	0.632008358027424

β_{il}					
0.325620674236780	0	0	0	0	0
0	0.237254825706663	0	0	0	0
0.000014278868889	0	0.324445943774684	0	0	0
0.000000008816565	0.034024265115088	0	0.195481644115112	0	0
0	0.000000001359460	0.000012509689649	0	0.227893014604489	0
0.033480821651945	0.001286838922731	0.058442764277772	0.041241903471131	0	0.205794987664170

Table 16 SSPRK(7,3)

α_{il} $\mathcal{C} = 3.740798731306490$

1.000000000000000	0	0	0	0	0	0
0.412429019730110	0.587570980269890	0	0	0	0	0
0.005800594241485	0	0.994199405758515	0	0	0	0
0.162485678538202	0.000000000270334	0	0.837514321191464	0	0	0
0.205239611567914	0.000000000554433	0.001461982584386	0	0.793298405293266	0	0
0.246951813330533	0.000686077138452	0.098274672761128	0.125080337194733	0	0.529007099575153	0
0.003515397992512	0.002051029751004	0.037621575915744	0.113733937331291	0.000552268540167	0	0.842525790469282

β_{il}

0.267322588523961	0	0	0	0	0	0
0	0.157070995387308	0	0	0	0	0
0.019051847781300	0	0.265771958656350	0	0	0	0
0.014327744686556	0.000000000072266	0	0.223886496266790	0	0	0
0.030979976588062	0.000000000148213	0.000390820968835	0	0.212066583174926	0	0
0.004054481853252	0.000183403916578	0.026271039908850	0.033436799512346	0	0.141415547205983	0
0.021050441338920	0.000548286582178	0.010057097058147	0.030403650530423	0.000147633855718	0	0.225226175206445

Table 17 SSPRK(8,3)

α_{il} $\mathcal{C} = 4.395231824884139$

1.000000000000000	0	0	0	0	0	0	0
0.108675201424538	0.891324798575462	0	0	0	0	0	0
0.008159777689219	0	0.991840222310781	0	0	0	0	0
0.000075204616622	0.000017473454611	0	0.999907321928768	0	0	0	0
0.275083494553101	0.013251614514063	0.333930523474093	0	0.377734367458743	0	0	0
0.172210423641858	0.067723791902171	0.031061316699451	0.018868041432255	0	0.710136426324266	0	0
0.155954681117895	0	0.000000000164948	0.000000000009768	0.000000000001983	0	0.844045318705406	0
0.021413729448041	0.000000000008708	0.000000000028479	0.072111559681400	0.109489249417096	0.046882143587611	0	0.750103317828665

β_{il}

0.227519284497891	0	0	0	0	0	0	0
0	0.202793580427116	0	0	0	0	0	0
0.026257801089696	0	0.225662777716378	0	0	0	0	0
0.000000000064040	0.000003975547891	0	0.227498198449436	0	0	0	0
0.006914605853513	0.003014997852682	0.075975633772832	0	0.085941853014477	0	0	0
0.000733937709508	0.015408468677066	0.007067048551022	0.004292843286543	0	0.161569731613186	0	0
0.000000000000089	0	0.000000000037529	0.000000000002222	0.000000000000451	0	0.192036586995649	0
0.024945755721405	0.000000000001981	0.000000000006480	0.016406770462739	0.024910915687589	0.010666591764781	0	0.170662970171872

Table 18 SSPRK(5,4)

α_{il}				$\mathcal{C} = 1.651549921326953$
1.000000000000000	0	0	0	0
0.261216512493821	0.738783487506179	0	0	0
0.623613752757655	0	0.376386247242345	0	0
0.444745181201454	0.120932584902288	0	0.434322233896258	0
0.213357715199957	0.209928473023448	0.063353148180384	0	0.513360663596212

β_{il}				
0.605491839566400	0	0	0	0
0	0.447327372891397	0	0	0
0.000000844149769	0	0.227898801230261	0	0
0.002856233144485	0.073223693296006	0	0.262978568366434	0
0.002362549760441	0.127109977308333	0.038359814234063	0	0.310835692561898

Table 19 SSPRK(6,4)

α_{il}					$\mathcal{C} = 2.227866058197466$
1.000000000000000	0	0	0	0	0
0.441581886978406	0.558418113021594	0	0	0	0
0.496140382330059	0	0.503859617669941	0	0	0
0.392013998230666	0.001687525300458	-0.000000000000000	0.606298476468875	0	0
0.016884674246355	0.000000050328214	0.000018549175549	0.000000000000000	0.983096726249882	0
0.128599802059752	0.150433518466544	0.179199506866483	0.173584325551242	0	0.368182847055979

β_{il}					
0.448860018455995	0	0	0	0	0
0	0.250651564517035	0	0	0	0
0.004050697317371	0	0.226162437286560	0	0	0
0.000000073512372	0.000757462637509	-0.000000000000000	0.272143145337661	0	0
0.000592927398846	0.000000022590323	0.000008325983279	0.000000000000000	0.441272814688551	0
0.000000009191468	0.067523591875293	0.080435493959395	0.077915063570602	0	0.165262559524728

Table 20 SSPRK(7,4)

α_{il}						$\mathcal{C} = 2.330275110889279$
1.000000000000000	0	0	0	0	0	0
0.277584603405600	0.722415396594400	0	0	0	0	0
0.528403304637363	0.018109310473034	0.453487384889603	0	0	0	0
0.363822566916605	0.025636760093079	0.000072932527637	0.610467740462679	0	0	0
0.080433061177282	0.000000001538366	0.000000000000020	0.000000000036824	0.919566937247508	0	0
0.305416318145737	0.017282647045059	0.214348299745317	0.001174022148498	0.003799138070873	0.457979574844515	0
0.112741543203136	0.042888410429255	0.185108001868376	0.000003952121250	0.230275526732661	0.110240916986851	0.318741648658470

β_{il}						
0.236998129331275	0	0	0	0	0	0
0.001205136607466	0.310012922173259	0	0	0	0	0
0.000000000029361	0.007771318668946	0.194606801046999	0	0	0	0
0.001612059039346	0.011001602331536	0.000031297818569	0.261972390131100	0	0	0
0.000000000027723	0.000000000660165	0.000000000000009	0.000000000015802	0.394617327778342	0	0
0.115125889382648	0.007416569384575	0.091984117559200	0.000503812679890	0.001630338861330	0.196534551952426	0
0.000102167855778	0.018404869978158	0.079436115076445	0.000001695989127	0.098819030275264	0.047308112450629	0.136782840433305

Table 21 SSPRK(8,4)

α_{il}							$\mathcal{C} = 3.542100748065554$
1.000000000000000	0	0	0	0	0	0	0
0.538569155333175	0.461430844666825	0	0	0	0	0	0
0.004485387460763	0	0.995514612539237	0	0	0	0	0
0.164495299288580	0.016875060685979	0	0.818629640025440	0	0	0	0
0.426933682982668	0.157047028197878	0.023164224070770	0.000000000000000	0.392855064748685	0	0	0
0.082083400476958	0.000000039091042	0.033974171137350	0.005505195713107	0.000000000000000	0.878437193581543	0	0
0.006736365648625	0.010581829625529	0.009353386191951	0.101886062556838	0.000023428364930	0	0.871418927612128	0
0.071115287415749	0.018677648343953	0.007902408660034	0.319384027162348	0.007121989995845	0.001631615692736	-0.000000000000000	0.574167022729334

β_{il}							
0.282318339066479	0	0	0	0	0	0	0
0.000000000000000	0.130270389660380	0	0	0	0	0	0
0.003963092203460	0	0.281052031928487	0	0	0	0	0
0.000038019518678	0.004764139104512	0	0.231114160282572	0	0	0	0
0.000019921336144	0.044337256156151	0.006539685265423	0.000000000000000	0.110910189373703	0	0	0
0.000000034006679	0.000000011036118	0.009591531566657	0.001554217709960	0.000000000000000	0.247998929466160	0	0
0.013159891155054	0.002987444564164	0.002640632454359	0.028764303955070	0.000006614257074	0	0.246017544274548	0
0.000000010647874	0.005273042658132	0.002230994887525	0.090167968072837	0.002010668386475	0.000460635032368	-0.000000000000000	0.162097880203691