

Solving $A\underline{x} = \underline{b}$ Using a Modified Conjugate Gradient Method Based on Roots of A

Paul F. Fischer¹ and Sigal Gottlieb²

Received January 23, 2001; accepted February 14, 2001

We consider the modified conjugate gradient procedure for solving $A\underline{x} = \underline{b}$ in which the approximation space is based upon the Krylov space associated with $A^{1/p}$ and \underline{b} , for any integer p . For the square-root MCG ($p = 2$) we establish a sharpened bound for the error at each iteration via Chebyshev polynomials in \sqrt{A} . We discuss the implications of the quickly accumulating effect of an error in $\sqrt{A}\underline{b}$ in the initial stage, and find an error bound even in the presence of such accumulating errors. Although this accumulation of errors may limit the usefulness of this method when $\sqrt{A}\underline{b}$ is unknown, it may still be successfully applied to a variety of small, "almost-SPD" problems, and can be used to jump-start the conjugate gradient method. Finally, we verify these theoretical results with numerical tests.

KEY WORDS: Modified conjugate gradient method; conjugate gradient method; Krylov space; convergence rate; stability.

1. INTRODUCTION

The modified conjugate gradient (MCG) method is based on the standard conjugate gradient (CG) method, which solves $A\underline{x} = \underline{b}$ (where $A \in \mathbb{R}^{n \times n}$ is symmetric positive definite) iteratively. At the k th iteration of CG, the solution \underline{x}_k comes from the Krylov space

$$V^k = \mathcal{K}_{A, \underline{b}}^k := \text{span}\{\underline{b}, A\underline{b}, A^2\underline{b}, \dots, A^{k-1}\underline{b}\}$$

¹ Math. and CS division, Argonne National Lab, Argonne, Illinois 60439; e-mail: fischer@mcs.anl.gov

² Department of Mathematics, UMASS-Dartmouth, North Dartmouth, Massachusetts 02747 and Division of Applied Mathematics, Box F, Brown University, Providence, Rhode Island 02912; e-mail: sg@cfm.brown.edu

In exact arithmetic, the CG method finds the best fit solution at each iteration [Lanczos (1952)]. That is, at the k th iteration, $\underline{x}_k \in V^k$ satisfies

$$\|\underline{x} - \underline{x}_k\|_A \leq \|\underline{x} - \underline{v}\|_A \quad \forall \underline{v} \in V^k \quad (1.1)$$

where the A -norm is given by $\|w\|_A = (w^T A w)^{1/2}$. As a result of this best fit property and the polynomial form of the Krylov space, the error can be bounded as [Birkhoff and Lynch (1984); Golub and Van Loan (1989)]:

$$\frac{\|\underline{x} - \underline{x}_k\|_A}{\|\underline{x} - \underline{x}_0\|_A} \leq 2 \left(\frac{\sqrt{\kappa_A} - 1}{\sqrt{\kappa_A} + 1} \right)^k \quad (1.2)$$

where κ_A is the condition number (the ratio of the largest eigenvalue to the smallest eigenvalue) of the matrix A . To improve upon the convergence rate, we must either change the condition number of the matrix A by preconditioning, or change the approximation space.

In Gottlieb and Fischer (1998) we presented a MCG method, which uses a finite-term recurrence and one multiplication by the matrix A per iteration to find the best-fit solution in the alternative approximation space, $\underline{x}_k \in \mathcal{H}_{\sqrt{A}, b}^k = \text{span}\{b, \sqrt{A}b, (\sqrt{A})^2 b, \dots, (\sqrt{A})^{k-1} b\}$. This approximation space suggests an (optimal) error bound

$$\frac{\|\underline{x} - \underline{x}_k\|_A}{\|\underline{x} - \underline{x}_0\|_A} \leq 2 \left(\frac{\sqrt{\kappa_M} - 1}{\sqrt{\kappa_M} + 1} \right)^k \quad (1.3)$$

which is based on κ_M , the condition number of $M = \sqrt{A}$. This error bound would be a significant improvement, since $\kappa_M = \sqrt{\kappa_A}$. With the usual initial guess $\underline{x}_0 = 0$, we previously obtained the error bound

$$\frac{\|\underline{x} - \underline{x}_k\|_A}{\|\underline{x}\|_A} \leq 2(k\kappa_M + 1) \left(\frac{\sqrt{\kappa_M} - 1}{\sqrt{\kappa_M} + 1} \right)^k \quad (1.4)$$

which is the optimal error bound multiplied by a factor which depends linearly on the condition number and the iterate. In this work, we sharpen the previous error bound to

$$\frac{\|\underline{x} - \underline{x}_k\|_A}{\|\underline{x}\|_A} \leq 2(2k + 1) \left(\frac{\sqrt{\kappa_M} - 1}{\sqrt{\kappa_M} + 1} \right)^k \quad (1.5)$$

which is a significant improvement, as the multiplicative factor now depends linearly only on the iterate, and there is no additional condition number dependence.

The idea is generalizable to the case where a series of vectors $\{\underline{b}, A^{1/p}\underline{b}, A^{2/p}\underline{b}, \dots, A^{(p-1)/p}\underline{b}\}$ are known initially. In that case, at each iteration the modified method finds the best fit in the Krylov space $V^k = \mathcal{K}_{A, \underline{b}}^k := \text{span}\{\underline{b}, A^{1/p}\underline{b}, A^{2/p}\underline{b}, \dots, A^{(k-1)/p}\underline{b}\}$. This is achieved with a finite term recurrence and only one matrix-vector multiplication per iteration. This approach will also be used to study the non-iterative approximation based on this series of vectors.

In theory, the MCG method could always be used instead of the CG method. In practice, however, the need for $A^{1/p}\underline{b}$ limits the use of this method. However, if $A^{1/p}\underline{b}$ can be approximated well and few iterations are needed, MCG may yield the optimal results. We will show analytically and numerically, (for $p=2$), that the error in this initial approximation builds up at each iteration and adversely affects the convergence rate. At worst, the method still converges as $1/p$ the rate of CG (i.e., every p th iteration is a CG iteration). This suggests that even where the MCG method is not the optimal choice, it may be useful for a few iterations, to lower the residual before switching to the CG method.

2. THE MODIFIED CONJUGATE GRADIENT METHOD AND THE EFFECT OF ERROR IN \sqrt{A}

We begin with a brief review of the construction of the MCG method detailed in Gottlieb and Fischer (1998). We define $\underline{x}_k \in V^k$ to be the solution vector at the k th iteration. Now let $\underline{e}_k = \underline{x} - \underline{x}_k$ be the error at the k th iteration, and let $\underline{r}_k = A(\underline{x} - \underline{x}_k)$ be the residual at the k th iteration. Usually we set $\underline{x}_0 = 0$ and so $\underline{r}_0 = \underline{b}$. Each \underline{x}_k is computed by

$$\underline{x}_k = \underline{x}_{k-1} + \alpha_k \underline{p}_k \quad (2.1)$$

For \underline{x}_k to be the best fit solution in V^k we require

$$\alpha_k = \frac{\underline{p}_k^T \underline{b}}{\underline{p}_k^T A \underline{p}_k} = \frac{\underline{p}_k^T \underline{r}_{k-1}}{\underline{p}_k^T A \underline{p}_k} \quad (2.2)$$

where $\underline{p}_k \in V^k$ and $\{\underline{p}_j\}$ form an A -orthogonal set (i.e., $\underline{p}_j^T A \underline{p}_m = 0$ for $j \neq m$). To find such a set, \underline{p}_k is chosen by picking a seed vector $\underline{v}_k \in V^k$ and using Gram-Schmidt orthogonalization with respect to $\{\underline{p}_j\}_1^{k-1}$.

$$\underline{p}_k = \underline{v}_k + \sum_{j=1}^{k-1} \beta_j \underline{p}_j \quad (2.3)$$

where

$$\beta_j = -\frac{\underline{p}_j^T A \underline{v}_k}{\underline{p}_j^T A \underline{p}_j} \quad (2.4)$$

The proposed square-root MCG (based on \sqrt{A} and \underline{b}) is obtained by taking the following sequence of seed vectors

$$\underline{v}_1 = \underline{r}_0 \quad (2.5)$$

$$\underline{v}_2 = \sqrt{A} \underline{r}_0 \quad (2.6)$$

$$\underline{v}_k = \underline{r}_{k-2}, \quad \text{for } k > 2 \quad (2.7)$$

which span the Krylov space $\mathcal{K}_{\sqrt{A}, \underline{b}}$, and reduce (2.3) to the finite term recurrence

$$\underline{p}_k = \underline{v}_k + \beta_{k-3} \underline{p}_{k-3} + \beta_{k-2} \underline{p}_{k-2} + \beta_{k-1} \underline{p}_{k-1} \quad (2.8)$$

What is remarkable about this method is that it has essentially the same complexity as CG (in terms of matrix-vector products), and achieves a superior convergence rate by choosing a candidate search direction that is *not* the steepest descent, that is, by choosing \underline{r}_{k-2} rather than the current residual \underline{r}_{k-1} .

If a sequence of vectors $\{A^{n/p} \underline{b}\}_{n=0}^{p-1}$ is known, the choice of seed vectors

$$\underline{v}_j = A^{(j-1)/p} \underline{b}, \quad \text{for } j = 1, \dots, p$$

$$\underline{v}_k = \underline{r}_{k-p}, \quad \text{for } k > p$$

will yield the Krylov space $\mathcal{K}_{A^{1/p}, \underline{b}}^k$, and a finite term recurrence

$$\underline{p}_k = \underline{v}_k + \sum_{j=1}^{2p-1} \beta_{k-j} \underline{p}_{k-j} \quad (2.9)$$

We expect this method to have an error bound which depends upon

$$\left(\frac{\sqrt{(\kappa_A)^{1/p}} - 1}{\sqrt{(\kappa_A)^{1/p}} + 1} \right)^k \quad (2.10)$$

This approach also suggests an error bound for a non-iterative approximation to the solution of $A \underline{x} = \underline{b}$. Given a sequence of vectors $\{A^{n/p} \underline{b}\}_{n=0}^{p-1}$, we can build the best-fit polynomial using the MCG procedure. This approximation will have an error bound asymptotically equal to (2.10) with $k = p - 1$.

The previous discussion assumed that we can readily find $\sqrt{A}\bar{b}$, or that some sequence of roots $\{A^{n/p}\}_{n=1}^{p-1}$ is known. Unfortunately, this is not always the case, and these quantities must usually be approximated. We now turn our attention to the case where $\sqrt{A}\bar{b}$ is approximated by $Q\bar{b}$, with some error $E = Q - \sqrt{A}$. The approximation space from which the (best-fit) k th iterate is taken is now $V^k = \text{span}\{\{\bar{b}, A\bar{b}, A^2\bar{b}, \dots, A^{[(k-1)/2]}\bar{b}\} \cup \{Q\bar{b}, A Q\bar{b}, A^2 Q\bar{b}, \dots, A^{[(k-2)/2]} Q\bar{b}\}\}$, where $[n]$ denotes the integer part of n . The k th iterate may be written as:

$$\begin{aligned} \bar{x}_k &= \sum_{j=0}^{[(k-1)/2]} c_j A^j \bar{b} + \sum_{j=0}^{[(k-2)/2]} d_j A^j Q\bar{b} \\ &= P_{k-1}(\sqrt{A}) \bar{b} + \tilde{P}_{[(k-2)/2]}(A) E\bar{b} \end{aligned}$$

Clearly, if the error $E=0$, then the first polynomial $P_{k-1}(\sqrt{A})\bar{b}$ is the MCG best fit polynomial. The second polynomial $\tilde{P}_{(k-1)/2-1}(A)$ can be understood as amplifying the error E introduced by approximating $\sqrt{A}\bar{b}$, and it is the odd part of the first polynomial. For the class of functions traditionally used to bound CG-type methods, we observe that the odd parts of such polynomials grow quickly with the order of the polynomial (see Fig. 2.1). This implies that the error introduced by approximating $\sqrt{A}\bar{b}$ will grow and destroy the convergence rate. However, this error will not grow without bound, since if we consider an even polynomial $P_{k-1}(\sqrt{A})$, the polynomial multiplying E would be zero. This approach is equivalent to the best fit even polynomial in the krylov space of CG so that even with a large error in approximating $\sqrt{A}\bar{b}$, the MCG method would converge at half the rate of CG, or equivalent to CG at every second iteration. This is an interesting guarantee on the convergence of MCG regardless of the initial approximation error. This error analysis suggests that MCG is useful where E very small and few iterations are needed, such as in the case of a matrix with few eigenvalues or where few iterations are used to reduce the residual before starting the CG method, or when the CG method stalls.

This approach is applicable to linear systems $B^T B \bar{x} = \bar{b}$ for which B is not symmetric, but is close enough to symmetric so that the simple (and cheap to compute) approximation $\sqrt{B^T B} \approx \frac{1}{2}(B + B^T)$ is highly accurate. For such systems, E is small, and if the number of iterations needed is small, the initial error will not have a chance to grow and diminish the convergence rate. Such cases will be discussed in the numerical section. In Gottlieb and Fischer (1998) we observed instability in the MCG method, even in the absence of an approximation to $\sqrt{A}\bar{b}$. This behavior was apparent more quickly in codes run with 32-bit precision, and diminished in 64-bit precision and 128-bit precision. We suggest that this instability

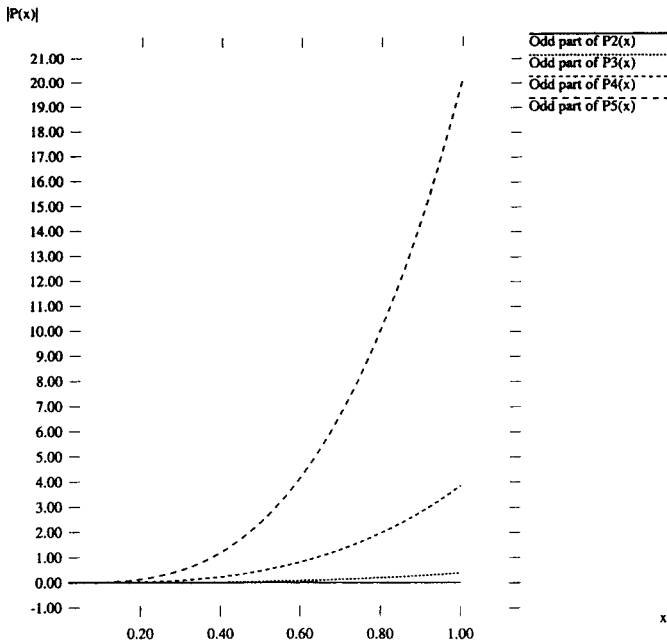


Fig. 2.1. The odd part of the polynomial $Q_{k+1}(x)$ used in the error analysis. This polynomial grows with its order, which implies that the error $E\bar{b}$ in the initial approximation of $\sqrt{A}\bar{b}$ will be amplified at each iteration.

can be explained as the result of machine-accuracy error in $\sqrt{A}\bar{b}$, and can be resolved only by using higher precision. This idea is validated by the fact that a series of numerical experiments in which E exists and is diminishing exhibits similar behavior. This behavior leads to a new view on preconditioning. A class of preconditioners which reduce the size of the relative error in $\sqrt{A}\bar{b}$ would help speed convergence. Traditional preconditioning aims to reduce the condition number of the iteration matrix. This new type of preconditioning would focus on making the approximation to \sqrt{A} more accurate. Specifically, for the case $A = B^T B$ discussed above, a preconditioner would make the matrix B more symmetric in some sense.

3. A SHARPER ERROR BOUND FOR THE MODIFIED METHOD

Given initial vectors $\{\bar{b}, \sqrt{A}\bar{b}\}$ the square-root MCG method finds the best fit in a readily computed approximation space, at a fixed cost per iteration. The “best fit” property implies that, in exact arithmetic, strict error bounds can be computed for the k th iterate, independent of the

particular algorithm used to compute \bar{x}_k . In this section we derive a sharper bound for the case where the approximation space is taken as the Krylov space $\mathcal{K}_{M, \bar{b}}^k$ associated with the symmetric positive definite (SPD) matrix $M := \sqrt{A}$.

The best-fit property implies that the k th iterate, $\bar{x}_k \in \mathcal{K}_{M, \bar{b}}^k$, is to satisfy (1.1):

$$\begin{aligned} \|\bar{x} - \bar{x}_k\|_A &\leq \|\bar{x} - \underline{v}\|_A && \forall \underline{v} \in \mathcal{K}_{M, \bar{b}}^k \\ &\leq \|\bar{x} - P_{k-1}(M)\bar{b}\|_A && \forall P_{k-1}(x) \in \mathbb{P}_{k-1}(x) \end{aligned} \quad (3.1)$$

where \mathbb{P}_{k-1} is the space of all polynomials of degree $k-1$ or less in the argument. Defining $\underline{r}_k = \bar{b} - A\bar{x}_k = A(\bar{x} - \bar{x}_k)$ as the residual at the k th iteration, we have $\|\bar{x} - \bar{x}_k\|_A = \|\underline{r}_k\|_{A^{-1}}$. The definition of M and (3.1) imply

For all $P_{k-1}(M) \in \mathbb{P}_{k-1}(M)$:

$$\begin{aligned} \|\bar{x} - \bar{x}_k\|_A &\leq \|\bar{b} - M^2 P_{k-1}(M)\bar{b}\|_{A^{-1}} \\ &\leq \|I - M^2 P_{k-1}(M)\|_{A^{-1}} \|\bar{b}\|_{A^{-1}} \\ &= \|I - M^2 P_{k-1}(M)\|_{A^{-1}} \|\bar{x}\|_A \end{aligned} \quad (3.2)$$

where the matrix norm, $\|\cdot\|_{A^{-1}}$, is the natural norm induced by the same vector norm. If M and A are any two SPD matrices which commute, and $Q(M)$ is any polynomial in M , a straightforward calculation reveals that $\|Q(M)\|_{A^{-1}} = \|Q(M)\|_2 = \rho(Q(M))$, where $\rho(Q)$ is spectral radius of Q . Consequently, an upper bound on $\|\bar{x} - \bar{x}_k\|_A$ can be derived by choosing a polynomial $Q(M) := I - M^2 P_{k-1}(M)$ which minimizes $\rho(Q)$. Denoting the eigenvalues of M by μ_i , where $0 < \mu_1 \leq \dots \leq \mu_n$, we have

$$\rho = \max_i |Q(\mu_i)| \leq \max_{\mu_1 \leq \mu \leq \mu_n} |Q(\mu)| \quad (3.3)$$

While the choice of P_{k-1} is arbitrary up to the maximal degree, $k-1$, the choice of $Q(x)$ is more restricted. Let $\mathbb{P}_{k+1}^{1,0}$ be the subset of \mathbb{P}_{k+1} defined by

$$\mathbb{P}_{k+1}^{1,0} = \{q : q(0) = 1; q'(0) = 0; q \in \mathbb{P}_{k+1}\} \quad (3.4)$$

Clearly, $Q \in \mathbb{P}_{k+1}^{1,0}$. Combining (3.2) and (3.3), one obtains

$$\frac{\|\bar{x} - \bar{x}_k\|_A}{\|\bar{x}\|_A} \leq \min_{Q \in \mathbb{P}_{k+1}^{1,0}} \max_{\mu_1 \leq \mu \leq \mu_n} |Q(\mu)| \quad (3.5)$$

The class of polynomials defined by (3.4) has been studied extensively by B. Fischer under the heading of Hermite kernel polynomials. Although (3.5) is not addressed directly, it is noted in B. Fischer (1996) that polynomials in $\mathbb{P}_{k+1}^{1,0}$ can be expressed as a linear combination of the same translated and scaled Chebyshev polynomials \hat{T}_k (defined below) that were used to derive (1.2). Although not the unique minimizer, the bound resulting from such a combination will be very close to optimal, as we now show.

We denote by \hat{T}_k the translated and scaled Chebyshev polynomial

$$\hat{T}_k(x) := \frac{T_k\left(\frac{\mu_n + \mu_1 - 2x}{\mu_n - \mu_1}\right)}{T_k\left(\frac{\mu_n + \mu_1}{\mu_n - \mu_1}\right)} \quad (3.6)$$

where $T_k(x)$ is the Chebyshev polynomial [e.g., Saad (1996)],

$$T_k(x) = \frac{1}{2}((x + \sqrt{x^2 - 1})^k + (x - \sqrt{x^2 - 1})^k), \quad |x| \geq 1 \quad (3.7)$$

\hat{T}_k satisfies the classic minimax problem on $[\mu_1, \mu_n]$:

$$m_k := \max_{\mu_1 \leq \mu \leq \mu_n} |\hat{T}_k(\mu)| = \min_{q \in \mathbb{P}_{k+1}^1} \max_{\mu_1 \leq \mu \leq \mu_n} |q(\mu)| \quad (3.8)$$

where \mathbb{P}_{k+1}^1 is the set of polynomials defined by

$$\mathbb{P}_{k+1}^1 = \{q : q(0) = 1; q \in \mathbb{P}_{k+1}\} \quad (3.9)$$

Defining $\sigma := (\mu_n + \mu_1)/(\mu_n - \mu_1)$, note that

$$m_k = \frac{1}{T_k(\sigma)} \quad (3.10)$$

Consider the polynomial $Q_{k+1}(x) = \alpha \hat{T}_{k+1}(x) + \beta \hat{T}_k(x)$. Since both $\hat{T}_{k+1}(x)$ and $\hat{T}_k(x)$ have the minimum possible extrema on $[\mu_1, \mu_n]$, this is clearly a reasonable starting point for solving the minimax problem (3.5). In order to satisfy the interpolatory constraints for $Q_{k+1} \in \mathbb{P}_{k+1}^{1,0}$, we must have

$$\alpha + \beta = 1$$

$$\alpha \hat{T}'_{k+1}(0) + \beta \hat{T}'_k(0) = 0$$

Solving for α and β yields

$$Q_{k+1}(x) = \frac{\hat{T}'_{k+1}(0) \hat{T}_k(x) - \hat{T}'_k(0) \hat{T}_{k+1}(x)}{\hat{T}'_{k+1}(0) - \hat{T}'_k(0)}$$

Note that $\hat{T}'_k(0)$ and $\hat{T}'_{k+1}(0)$ are of the same sign, whereas $\hat{T}_{k+1}(\mu_n)$ and $\hat{T}_k(\mu_n)$ are of opposite sign. Thus, we have

$$\max_{\mu_1 \leq \mu \leq \mu_n} |Q_{k+1}(\mu)| = \frac{\hat{T}'_k(0) m_{k+1} + \hat{T}'_{k+1}(0) m_k}{\hat{T}'_{k+1}(0) - \hat{T}'_k(0)}$$

Using (3.6) and (3.10), one obtains the bound

$$\max_{\mu_1 \leq \mu \leq \mu_n} |Q_{k+1}(\mu)| = \frac{T'_{k+1} + T'_k}{T_k T'_{k+1} - T_{k+1} T'_k} \quad (3.11)$$

where the argument of the Chebyshev polynomials and their derivatives is taken to be σ .

To compare this bound to the original CG result (1.2), we recast (3.11) in terms of $\kappa_M := \mu_n/\mu_1$, the condition number of M . Defining $a := \sigma + \sqrt{\sigma^2 - 1}$ and $b := \sigma - \sqrt{\sigma^2 - 1}$, we note the following identities:

$$\begin{aligned} a &= \frac{\sqrt{\kappa_M + 1}}{\sqrt{\kappa_M - 1}} \\ a \cdot b &= 1 \\ a > b > 0 \\ T_k(\sigma) &= \frac{1}{2} (a^k + b^k) \\ T'_k(\sigma) &= \frac{k}{a - b} (a^k - b^k) \end{aligned} \quad (3.12)$$

The denominator of (3.11) is then:

$$\begin{aligned} &T_{k+1} T'_k - T_k T'_{k+1} \\ &= \left| \frac{k+1}{2(a-b)} (a^k + b^k)(a^{k+1} - b^{k+1}) - \frac{k}{2(a-b)} (a^{k+1} + b^{k+1})(a^k - b^k) \right| \\ &= k + \frac{1}{2} + \frac{a^{2k+1} - b^{2k+1}}{2(a-b)} \end{aligned} \quad (3.13)$$

while the numerator becomes

$$T'_k + T'_{k+1} = \frac{k}{2(a-b)}(a^k - b^k) + \frac{k+1}{2(a-b)}(a^{k+1} - b^{k+1}) \quad (3.14)$$

Combining (3.13) and (3.14) yields

$$\begin{aligned} \max_{\mu_1 \leq \mu \leq \mu_n} |Q_{k+1}(\mu)| &= \frac{k(a^k - b^k) + (k+1)(a^{k+1} - b^{k+1})}{(a-b)(k + \frac{1}{2}) + \frac{1}{2}(a^{2k+1} - b^{2k+1})} \\ &\leq \frac{k(a^k - b^k) + (k+1)(a^{k+1} - b^{k+1})}{\frac{1}{2}(a^{2k+1} - b^{2k+1})} \\ &= \frac{2k(a^k - b^k)}{(a^{2k+1} - b^{2k+1})} + \frac{2(k+1)(a^{k+1} - b^{k+1})}{(a^{2k+1} - b^{2k+1})} \\ &= \frac{2k(a^k - b^k)}{a^{k+1}a^k - a^{k+1}b^k + (a^{k+1}b^k - b^{k+1}b^k)} \\ &\quad + \frac{2(k+1)(a^{k+1} - b^{k+1})}{a^k a^{k+1} - a^k b^{k+1} + (a^k b^{k+1} - b^k b^{k+1})} \\ &\leq \frac{2k(a^k - b^k)}{a^{k+1}a^k - a^{k+1}b^k} + \frac{2(k+1)(a^{k+1} - b^{k+1})}{a^k a^{k+1} - a^k b^{k+1}} \\ &= \frac{2k}{a^{k+1}} + \frac{2(k+1)}{a^k} \\ &\leq \frac{2(2k+1)}{a^k} \end{aligned} \quad (3.15)$$

Taking (3.15) in conjunction with the first of the identities (3.12), we obtain the desired result

$$\frac{\|x - x_k\|_A}{\|x\|_A} \leq 2(2k+1) \left(\frac{\sqrt{\kappa_M} - 1}{\sqrt{\kappa_M} + 1} \right)^k \quad (3.16)$$

Although this bound has an extra factor of $(2k+1)$ not present in (1.2), the fact that it is based upon $\kappa_M = \sqrt{\kappa_A}$ implies that the modified approach should yield a much better convergence rate than the standard CG algorithm. A comparison of this new error bound (3.16), the previous error bound (1.4), the optimal error bound (1.3) and the CG error bound (1.2) is shown in Fig. 3.1.

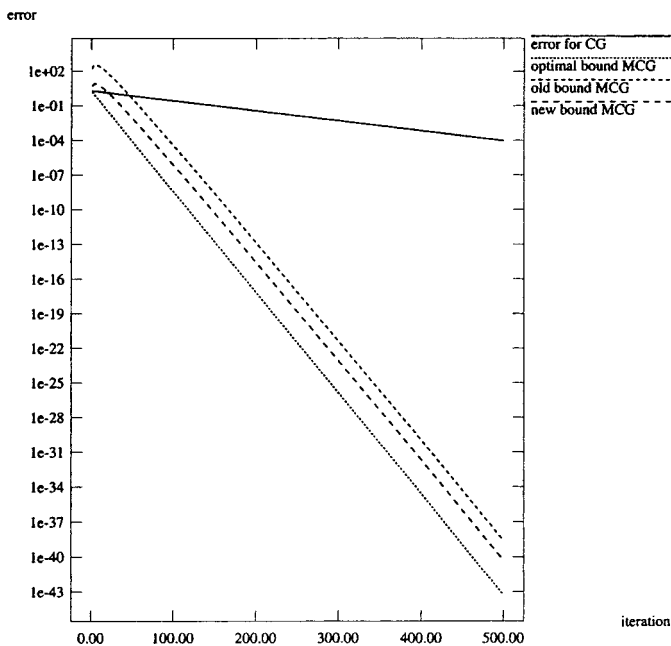


Fig. 3.1. Comparison of the CG and MCG error bounds. The graph shows the $\log(\text{error})$ as a function of the number of iterations. Clearly, the MCG bounds show the error decaying at a rate which is a significant improvement over the error decay rate associated with CG.

4. NUMERICAL EXPERIMENTS

Example 1. The MCG and CG methods are applied to the problem $A\underline{x} = \underline{b}$ where $A \in \mathbb{R}^{1600 \times 1600}$ is the two dimensional Laplacian operator given by second-order finite differences on a regular array of points:

$$(Av)_{i,j} = 4v_{i,j} - v_{i+1,j} - v_{i-1,j} - v_{i,j+1} - v_{i,j-1}$$

with corresponding Dirichlet boundary conditions. The right hand side \underline{b} is given by

$$b_j = 100 \sin(100 \cos(j))$$

In this case, the square-root of the matrix A is unknown, but is approximated to a high degree of accuracy, using a method developed by van der Vorst (1987). The performance of the MCG method is compared to that of the standard CG. The initial behavior of MCG is exactly as predicted (Fig. 4.1, right), following along the error-bound almost exactly.

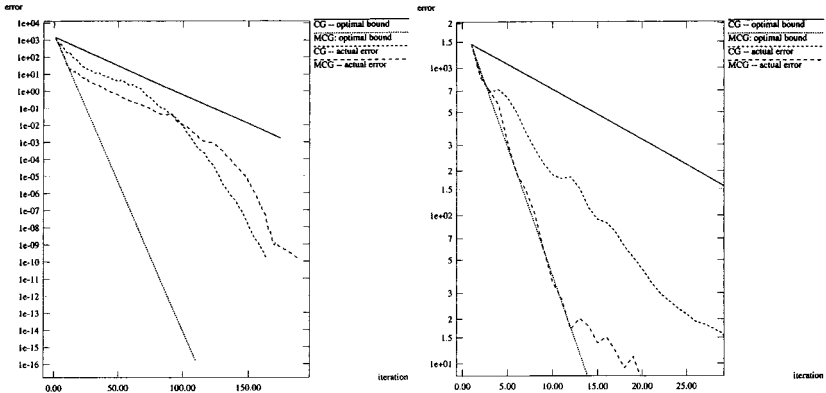


Fig. 4.1. This is a comparison of the performance of MCG and CG for Example 1, and the theoretical error bounds of MCG and CG. the initial convergence of MCG matches perfectly with the optimal error bound initially (right). MCG initially decays as predicted by the error bound and outperforms CG, but later (left) the error associated with the approximation of $\sqrt{A} \underline{b}$ grows and causes convergence to slow down. near 100 iterations MCG begins to converge at half the rate of CG, however, at no point does the error decay at twice the rate of the CG error bound.

Example 2. MCG and CG are applied to a series of problems of the form

$$B^T B \underline{x} = \underline{b}$$

where B are 150×150 matrices of the form:

$$B = \begin{pmatrix} 2.5 & -1 + \varepsilon & 0 & 0 & 0 & \dots & 0 \\ -1 & 2.5 & -1 + \varepsilon & 0 & 0 & \dots & 0 \\ 0 & -1 & 2.5 & -1 + \varepsilon & 0 & \dots & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & 0 & -1 & 2.5 & -1 + \varepsilon \\ 0 & 0 & 0 & 0 & 0 & -1 & 2.5 \end{pmatrix}$$

and ε , depending on the case, takes values $10^{-8} \leq \varepsilon \leq 10^{-2}$. In all cases, \underline{b} is given by $b_j = \sin(100 \cos(100j))$. The square-root initial vector $\sqrt{A} \underline{b}$ is approximated by

$$\sqrt{B^T B} \underline{b} \approx \frac{B^T + B}{2} \underline{b}$$

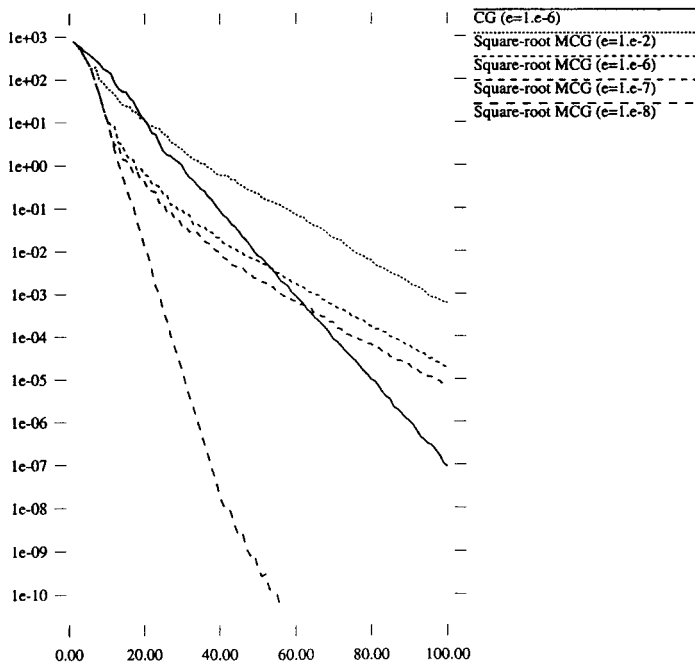


Fig. 4.2. CG and the square-root MCG were applied as in Example 2, for a few different values of ε . CG behaved almost exactly the same regardless of the value of ε , so one representative case is shown. The behavior of MCG differed widely depending on the value of ε . When $\varepsilon = 10^{-2}$, the error buildup diminished the convergence rate after very few iterations. At the level of $\varepsilon = 10^{-6}$ and $\varepsilon = 10^{-7}$, the initial convergence of MCG is a significant improvement for the first 60 or so iterations, at which point there is a crossover, and CG is better than MCG. However, for $\varepsilon = 10^{-8}$ MCG is a significant improvement over CG.

Figure 4.2 shows the effect of the error in the square-root by varying ε . When ε is small, this approximation is almost exact, and the error-vector E is small. When ε grows, E grows correspondingly. To limit the effect of round-off error, the MCG codes were run in 128-bit precision. The effect is remarkably as we expect. The MCG method converges significantly faster than CG when ε is very small. As ε grows, we see the MCG method exhibiting slowed convergence at an earlier iteration number. This diminished convergence causes MCG to converge slower than CG in the cases where $\varepsilon \geq 10^{-7}$.

Example 3. The cube-root MCG, square-root MCG and CG were used to solve $A\bar{x} = \bar{b}$ where A is a 900×900 diagonal matrix

$$A = \begin{pmatrix} \lambda_1 & 0 & 0 & 0 & \cdots & 0 \\ 0 & \lambda_2 & 0 & 0 & \cdots & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & 0 & 0 & \lambda_{900} \end{pmatrix}$$

with λ_i given by

(a)

$$\lambda_1 = 0.034, \quad \lambda_2 = 0.082, \quad \lambda_3 = 0.127, \quad \lambda_4 = 0.155, \quad \lambda_5 = 0.19$$

$$\lambda_i = 0.2 + \frac{i-5}{895} \quad i = 6, 900$$

(b)

$$\lambda_1 = 214.827, \quad \lambda_2 = 57.4368, \quad \lambda_3 = 48.5554, \quad \lambda_4 = 35.0624$$

$$\lambda_5 = 27.3633, \quad \lambda_6 = 21.8722, \quad \lambda_7 = 17.7489$$

$$\lambda_i = 1.0 + 15.6624 \frac{i-8}{892} \quad i = 8, 900$$

Figure 4.3 compares the standard CG method, the square-root MCG and the cube-root MCG. The performance is as predicted, with the cube-root method converging fastest, the square-root MCG next and the standard method last. Although this was an idealized case, in which the matrix was a diagonal matrix and the square- and cube-roots were calculated directly, and the MCG codes were run with 128-bit precision, both MCG methods eventually suffer from instability after the error was below 10^{-11} . This did not seem to affect convergence, but if 128-bit precision was not used, the effect of round-off error would be seen much sooner. These two cases show that the ideal behavior of MCG is indeed as predicted.

5. CONCLUSIONS

We generalize the MCG method to cases where we have a available a sequence of vectors $\{\underline{b}, A^{1/p}\underline{b}, A^{2/p}\underline{b}, \dots, A^{(p-1)/p}\underline{b}\}$, and discuss the approximation properties of such a method, as well as that of the non-iterative approximation based on this sequence. A sharpened bound is obtained for the square-root case, as well as an analysis of the effect of an error in the initial approximation of $\sqrt{A}\underline{b}$. We suspect that this effect may be a cause

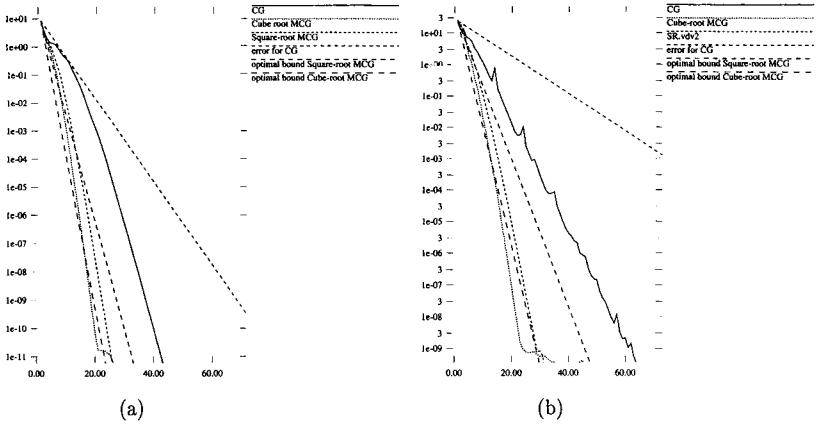


Fig. 4.3. The cube-root and square-root MCG methods are compared to CG for the two van der Vorst matrices (a) and (b) in Example 3, respectively. The MCG methods initially exceed the optimal bound, but later converge faster, verifying the claim that the bound will have some iteration-related term multiplied to the optima-bound term we're using. The diminished convergence; which we explain as the effect of round-off level error in the \sqrt{A} is not visible, as it occurs after the residual was cut to below 10^{-11} , but it does occur a little later.

of instability in the MCG method, and suggest that a new approach to preconditioning may resolve this problem. This method is still applicable to cases in which we are solving $B^T Bx = b$ where B is close to symmetric, and the number of iterations needed is small. In the numerical experiments, we verify that the initial convergence rate of the square-root ($p = 2$) and cube-root ($p = 3$) MCG is a significant improvement over CG, converging at a rate which depends upon with the p th-root of the condition number. We also observe, as predicted, the accumulating effect of an error in \sqrt{A} at the initial stage, and at worst, equivalence of the convergence rate of MCG to CG at every second iteration.

ACKNOWLEDGEMENTS

This was supported by the Mathematical, Information, and Computational Sciences Division subprogram of the Office of Advanced Scientific Computing Research, U.S. Department of Energy, under Contract W-31-109-Eng-38.

REFERENCES

Birkhoff, G., and Lynch, R. E. (1984). *Numerical Solution of Elliptic Problems*, SIAM, Philadelphia.

- Fischer, B. (1996). *Polynomial Based Iteration Methods for Symmetric Linear Systems*, Wiley-Teubner, Chichester, Stuttgart.
- Golub, G. H., and Van Loan, C. H. (1989). *Matrix Computations*, The John Hopkins University Press, Baltimore.
- Gottlieb, S., and Fischer, P. F. (1998). A modified conjugate gradient method for the solution of $Ax = b$ based upon b and $A^{1/2}b$. *J. of Sci. Comput.* **13**(2), 173–183.
- Lanczos, C. (1952). Solution of systems of linear equations by minimized iterations. *J. of Research of the National Bureau of Standards* **49**, 33–53.
- O’Leary, D. P. (1980). The block conjugate gradient algorithm and related methods. *Linear Algebra and its Appl.* **29**, 293–322.
- Simon, H. D. (1984). The Lanczos method with partial reorthogonalization. *Math. of Comp.* **42**, 115–142.
- Van der Vorst, H. A. (1987). An iterative solution method for solving $f(A)x = \underline{b}$, using Krylov subspace information obtained for the symmetric positive definite matrix A . *J. of Comp. and Appl. Math.* **18**, 249–263.

Strong Stability Preserving Properties of Runge–Kutta Time Discretization Methods for Linear Constant Coefficient Operators

Sigal Gottlieb¹ and Lee-Ad J. Gottlieb¹

Received August 21, 2001; accepted (in revised form) January 11, 2002

Strong stability preserving (SSP) high order Runge–Kutta time discretizations were developed for use with semi-discrete method of lines approximations of hyperbolic partial differential equations, and have proven useful in many other applications. These high order time discretization methods preserve the strong stability properties of first order explicit Euler time stepping. In this paper we analyze the SSP properties of Runge–Kutta methods for the ordinary differential equation $u_t = Lu$ where L is a linear operator. We present optimal SSP Runge–Kutta methods as well as a bound on the optimal timestep restriction. Furthermore, we extend the class of SSP Runge–Kutta methods for linear operators to include the case of time dependent boundary conditions, or a time dependent forcing term.

KEY WORDS: Strong stability preserving; Runge–Kutta methods; high order accuracy; time discretization.

1. INTRODUCTION

1.1. The History of SSP Methods

In solving time dependent hyperbolic Partial Differential Equations (PDEs) it is common practice to first discretize the spatial variables to obtain a semi-discrete method of lines scheme. This is then an Ordinary Differential Equation (ODE) system in the time variable which can be discretized by an

¹Department of Mathematics, University of Massachusetts at Dartmouth, Dartmouth, Massachusetts 02747 and Division of Applied Mathematics, Brown University, Providence, Rhode Island 02912. E-mail: sgottlieb@umassd.edu

ODE solver. The simplest such ODE solver is the forward-Euler method and it is used widely for analysis of the nonlinear stability properties of the spatial discretization. The nonlinear stability properties are essential, since hyperbolic problems typically have discontinuous solutions and a stronger measure than linear stability is thus required. However, while forward-Euler is ideal for analysis of the stability properties of a given spatial discretization, it is only first order accurate. In practice, high order time discretizations which preserve all the stability properties of forward-Euler, are needed.

In [15, 14, 4 and 5] high order strong stability preserving (SSP) time discretization methods for the semi-discrete method of lines approximations of PDEs were developed. These methods are derived by assuming that the first order forward-Euler time discretization of the method of lines ODE is strongly stable under a certain norm, when the time step Δt is suitably restricted, and then finding higher order time discretizations that maintain strong stability for the same norm, perhaps under a different time step restriction.

SSP Runge Kutta methods were first considered for the solution of the hyperbolic equation

$$u_t + f(u)_x = 0 \quad (1.1)$$

where the spatial derivative, $f(u)_x$, is discretized by a TVD finite difference or finite element approximation denoted $-L(u)$ ([9, 12, 18, 3, 10] and [19]). In the process of discretizing, we have a spatial mesh made up of points denoted x_j and a temporal mesh of points denoted t^n . When discussing the fully discretized solution, we use the notation u_j^n to mean the approximation to the exact solution $u(x_j, t^n)$, and the corresponding vector u^n containing all the spatial information at a given time is given componentwise by $u^n = [u_j^n]$. The exact spatial discretization $L(u)$ above is irrelevant, as long as it has the property that when it is combined with the first order forward-Euler time discretization,

$$u^{n+1} = u^n + \Delta t L(u^n) \quad (1.2)$$

the Total Variation (TV) of the one-dimensional discrete solution u^n does not increase in time, i.e., the following, so called Total Variation Diminishing (TVD) property, holds

$$\text{TV}(u^{n+1}) \leq \text{TV}(u^n), \quad \text{TV}(u^n) := \sum_j |u_{j+1}^n - u_j^n| \quad (1.3)$$

for a sufficiently small time step Δt dictated by the CFL condition (see [1, 8]),

$$\Delta t \leq \Delta t_{\text{FE}} \quad (1.4)$$

Here, Δt_{FE} is the largest allowable step size that will guarantee that the stability property above will hold for forward-Euler with the given PDE and spatial discretization (see [14]).

The objective of the high order SSP time discretization is to maintain the strong stability property (1.3) while achieving higher order accuracy in time, perhaps with a modified CFL restriction (measured here with a CFL coefficient, c)

$$\Delta t \leq c \Delta t_{\text{FE}} \quad (1.5)$$

Numerical evidence presented in [4] demonstrated that oscillations may occur when using a linearly stable, high-order method which lacks the strong stability property, even if the same spatial discretization is TVD when combined with the first-order forward-Euler time-discretization. This illustrates that it is safer to use a SSP time discretization for solving hyperbolic problems. After all, SSP methods have the extra assurance of provable stability and in many cases do not increase the computational cost. In particular, SSP methods up to (and including) third order for ODEs with nonlinear operators L , and all SSP methods for ODEs with linear constant-coefficient operators do not require any additional stages or function evaluations [5].

In the initial development of these methods, ([15] and [14]) the relevant norm was the total variation norm: the forward-Euler time discretization of the method of lines ODE was assumed TVD, hence these methods were called TVD time discretizations. In fact, the essence of this class of high order time discretizations lies in its ability to maintain the strong stability in the same norm as the first order forward-Euler version, regardless of what this norm is, hence “strong stability preserving (SSP) time discretization” is a more suitable term which was first adopted in [5]. Additionally, since SSP methods (as we show below) are convex combinations of the first-order Euler method, any convex function satisfied by forward-Euler will be preserved by such high-order time discretizations. Thus, the SSP property is useful in a wide variety of applications. SSP Runge Kutta methods can be used whenever a method is needed which preserves any norm or convex-function property of forward-Euler. Also, although these methods were developed for use with nonlinear stability properties, they are equally useful in cases where the relevant operator is linear, and where

linear norm properties are studied. In this paper we will study the properties of SSP Runge–Kutta methods for linear constant-coefficient operators.

1.2. SSP Runge–Kutta Methods

In [15], a general m stage Runge–Kutta method for

$$u_t = L(u) \quad (1.6)$$

is written in the form:

$$\begin{aligned} u^{(0)} &= u^n, \\ u^{(i)} &= \sum_{k=0}^{i-1} (\alpha_{i,k} u^{(k)} + \Delta t \beta_{i,k} L(u^{(k)})), \quad \alpha_{i,k} \geq 0, \quad i = 1, \dots, m \\ u^{n+1} &= u^{(m)} \end{aligned} \quad (1.7)$$

This way of writing the method has become standard for SSP purposes (e.g., [14, 15, 4, 5]), and was shown ([15]) to be equivalent to the classical Runge–Kutta methods as written in [2]. The restriction on the coefficients $\alpha_{i,k}$ allows the SSP property to be achieved ([14]). Clearly, if all the coefficients $\beta_{i,k}$ are nonnegative ($\beta_{i,k} \geq 0$), and the consistency requirement

$$\sum_{k=0}^{i-1} \alpha_{i,k} = 1$$

is satisfied for any i , it follows that the intermediate stages in (1.7), $u^{(i)}$, amount to convex combinations of forward-Euler steps, with Δt replaced by $\frac{\beta_{i,k}}{\alpha_{i,k}} \Delta t$. We thus conclude

Lemma 1.1 [15]. If the forward-Euler method (1.2) is strongly stable under the CFL restriction (1.4), $\|u^n + \Delta t L(u^n)\| \leq \|u^n\|$, then the Runge–Kutta method (1.7) with $\beta_{i,k} \geq 0$, and $\beta_{i,k} = 0$ whenever $\alpha_{i,k} = 0$, is SSP, $\|u^{n+1}\| \leq \|u^n\|$, provided the following CFL restriction (1.5) is fulfilled,

$$\Delta t \leq c \Delta t_{\text{FE}}, \quad c = \min_{i,k} \frac{\alpha_{i,k}}{\beta_{i,k}} \quad \forall \beta_{i,k} \neq 0 \quad (1.8)$$

If some of the $\beta_{i,k}$'s are negative, we need to introduce an associated operator \tilde{L} corresponding to stepping *backward* in time. The requirement for \tilde{L} is that it approximates the same spatial derivative(s) as L , but that

the strong stability property holds $\|u^{n+1}\| \leq \|u^n\|$, (either with respect to the TV or another relevant norm), for the first order Euler scheme, solved backward in time, i.e.,

$$u^{n+1} = u^n - \Delta t \tilde{L}(u^n) \quad (1.9)$$

This can be achieved, for hyperbolic conservation laws, by solving the time-negative version of (1.1),

$$u_t = f(u)_x \quad (1.10)$$

Numerically, the only difference is the change of upwind direction. Clearly, \tilde{L} can be computed with the same cost as that of computing L . We then have the following lemma.

Lemma 1.2 [15]. If the forward-Euler method combined with the spatial discretization L in (1.2) is strongly stable under the CFL restriction (1.4), $\|u^n + \Delta t L(u^n)\| \leq \|u^n\|$, and if Euler's method solved backward in time in combination with the spatial discretization \tilde{L} in (1.9) is also strongly stable under the CFL restriction (1.4), $\|u^n - \Delta t \tilde{L}(u^n)\| \leq \|u^n\|$, then the Runge–Kutta method (1.7) is SSP, i.e., $\|u^{n+1}\| \leq \|u^n\|$, under the CFL restriction (1.5),

$$\Delta t \leq c \Delta t_{\text{FE}}, \quad c = \min_{i,k} \frac{\alpha_{i,k}}{|\beta_{i,k}|}, \quad \forall \beta_{i,k} \neq 0 \quad (1.11)$$

provided $\beta_{i,k} = 0$ whenever $\alpha_{i,k} = 0$, and $\beta_{i,k}L$ is replaced by $\beta_{i,k}\tilde{L}$ whenever $\beta_{i,k}$ is negative.

Notice that, if for the same k , both $L(u^{(k)})$ and $\tilde{L}(u^{(k)})$ must be computed, the cost as well as storage requirement for this k is doubled. For this reason, we would like to avoid negative $\beta_{i,k}$ as much as possible. In [4] it was shown any four-stage fourth-order Runge–Kutta method for a nonlinear ODE will have at least one negative coefficient. In [17] it was shown that any Runge–Kutta method of fifth order or above for a nonlinear ODE will have at least one negative coefficient. Thus, we realize that for Runge–Kutta methods for nonlinear ODEs, negative coefficients would have to be considered. This is not, however, the case for Runge–Kutta methods for linear ODEs. In the linear constant coefficient case, we may have all nonnegative coefficients [5], and we proceed by discussing this case.

2. LINEAR CONSTANT COEFFICIENT SSP RUNGE–KUTTA METHODS OF ARBITRARY ORDER

Although SSP methods were developed for use with nonlinear stability properties, they are equally useful in cases where the relevant operator is linear, and where linear norm properties are studied. For example, SSP methods are useful where weighted L^2 higher order discretizations of spectral schemes are discussed ([7, 5, 11]). In [5] we found optimal N stage, N th order SSP Runge–Kutta methods of arbitrary order of accuracy for *linear* ODEs suitable for solving PDEs with linear spatial discretizations. Such methods have optimal CFL number $c = 1$. Raising the CFL number at the expense of adding another stage is an idea that was tried in [14]. In parallel to this work, S. Ruuth and R. Spiteri have studied this approach for nonlinear methods [16] and linear methods [17]. In this section, we consider linear SSP Runge–Kutta methods which have more stages than necessary for their order. This additional freedom allows for a higher CFL number. We present a bound on the optimal CFL number associated with an m stage, N th order method. We then present some methods which are optimal in terms of the CFL restriction.

2.1. Useful Properties of the Linear SSP Runge–Kutta Method

In [5] we constructed a class of optimal (in the sense of CFL number) SSP Runge–Kutta methods of any order for the ODE (1.6) where L is *linear* and *time invariant*. With a linear L being realized as finite dimensional matrix we denote $L(u) = Lu$.

The method (1.7) may be rewritten as

$$u^{(i)} = \left(1 + \sum_{k=0}^{i-1} A_{i,k} (\Delta t L)^{k+1} \right) u^{(0)}, \quad i = 1, \dots, m \quad (2.1)$$

where

$$\begin{aligned} A_{1,0} &= \beta_{1,0}, & A_{i,0} &= \sum_{k=1}^{i-1} \alpha_{i,k} A_{k,0} + \sum_{k=0}^{i-1} \beta_{i,k} \\ A_{i,k} &= \sum_{j=k+1}^{i-1} \alpha_{i,j} A_{j,k} + \sum_{j=k}^{i-1} \beta_{i,j} A_{j,k-1}, & k &= 1, \dots, i-1 \end{aligned}$$

A method of this type will be SSP for a sufficiently small time step Δt , if all the coefficients $\alpha_{i,k}$ and $\beta_{i,k}$ are nonnegative. The CFL number (c in (1.8)) associated with this method can be written as $c = \frac{1}{\mu}$ where

$\mu = \max_{i,k} \frac{\beta_{i,k}}{\alpha_{i,k}}$. To facilitate the analysis of the optimal CFL number, we introduce $\mu_{i,k} = \frac{\beta_{i,k}}{\alpha_{i,k}}$. We can make this definition, since the SSP conditions above require that $\beta_{i,k} = 0$ whenever $\alpha_{i,k} = 0$. The following lemmas determine a bound on the relative size of each $A_{i,k}$, which depends on μ . These lemmas will later be used to get a bound on the optimal CFL number.

Lemma 2.1. For any method written in the form (2.1) above,

$$A_{M,0} \leq M\mu$$

for any $1 \leq M \leq m$, where $\mu = \max_{i,k} \frac{\beta_{i,k}}{\alpha_{i,k}}$.

Proof. Consider that

$$A_{1,0} = \beta_{1,0} = \frac{\beta_{1,0}}{\alpha_{1,0}} = \mu_{1,0} \leq \mu$$

Now proceed by induction: Assume $A_{j,0} \leq j\mu \forall j = 1, \dots, M-1$ then

$$\begin{aligned} A_{M,0} &= \sum_{j=1}^{M-1} \alpha_{M,j} A_{j,0} + \sum_{j=0}^{M-1} \beta_{M,j} \\ &\leq (M-1)\mu \sum_{j=1}^{M-1} \alpha_{M,j} + \mu \sum_{j=0}^{M-1} \alpha_{M,j} \\ &\leq M\mu \end{aligned} \quad \square$$

Lemma 2.2. For any method written in the form (2.1) above,

$$A_{M,1} \leq \frac{M-1}{2} \mu A_{M,0}$$

for any $1 \leq M \leq m$, where $\mu = \max_{i,k} \frac{\beta_{i,k}}{\alpha_{i,k}}$.

Proof.

$$\begin{aligned} A_{2,1} &= \beta_{2,1} A_{1,0} = \frac{1}{2} (\beta_{2,1} A_{1,0} + \beta_{2,1} \beta_{1,0}) \\ &= \frac{1}{2} (\mu_{2,1} \alpha_{2,1} A_{1,0} + \beta_{2,1} \mu_{1,0}) \\ &\leq \frac{1}{2} (\mu_{2,1} \alpha_{2,1} A_{1,0} + \beta_{2,1} \mu_{1,0} + \mu \beta_{2,0}) \\ &\leq \frac{1}{2} \mu (\alpha_{2,1} A_{1,0} + \beta_{2,1} + \beta_{2,0}) \\ &= \frac{1}{2} \mu A_{2,0} \end{aligned}$$

Proceed by induction: assume

$$A_{j,1} \leq \frac{j-1}{2} \mu A_{j,0} \quad \text{for } 2 \leq j \leq M-1$$

then

$$\begin{aligned} A_{M,1} &= \sum_{j=2}^{M-1} \alpha_{M,j} A_{j,1} + \sum_{j=1}^{M-1} \beta_{M,j} A_{j,0} \\ &\leq \sum_{j=2}^{M-1} \alpha_{M,j} \frac{j-1}{2} \mu A_{j,0} + \sum_{j=1}^{M-1} \mu_{M,j} \alpha_{M,j} A_{j,0} \\ &\leq \frac{M-2}{2} \mu \sum_{j=1}^{M-1} \alpha_{M,j} A_{j,0} + \sum_{j=1}^{M-1} \mu_{M,j} \alpha_{M,j} A_{j,0} \end{aligned}$$

where the zero term $\alpha_{M,j} \frac{j-1}{2} \mu A_{j,0}$ for $j=1$ was added in the first summation.

$$\begin{aligned} &= \frac{M-1}{2} \mu \sum_{j=1}^{M-1} \alpha_{M,j} A_{j,0} + \sum_{j=1}^{M-1} \left(\mu_{M,j} - \frac{1}{2} \mu \right) \alpha_{M,j} A_{j,0} \\ &\leq \frac{M-1}{2} \mu \sum_{j=1}^{M-1} \alpha_{M,j} A_{j,0} + \sum_{j=1}^{M-1} \left(\mu_{M,j} - \frac{1}{2} \mu_{M,j} \right) \alpha_{M,j} A_{j,0} \\ &= \frac{M-1}{2} \mu \sum_{j=1}^{M-1} \alpha_{M,j} A_{j,0} + \sum_{j=1}^{M-1} \frac{1}{2} \mu_{M,j} \alpha_{M,j} A_{j,0} \\ &\leq \frac{M-1}{2} \mu \sum_{j=1}^{M-1} \alpha_{M,j} A_{j,0} + \frac{1}{2} (M-1) \mu \sum_{j=1}^{M-1} \mu_{M,j} \alpha_{M,j} \\ &= \frac{M-1}{2} \mu \sum_{j=1}^{M-1} \alpha_{M,j} A_{j,0} + \frac{1}{2} (M-1) \mu \sum_{j=1}^{M-1} \beta_{M,j} \\ &\leq \frac{M-1}{2} \mu \left(\sum_{j=1}^{M-1} \alpha_{M,j} A_{j,0} + \sum_{j=0}^{M-1} \beta_{M,j} \right) \quad \text{where we added} \end{aligned}$$

the nonnegative quantity $\frac{M-1}{2} \mu \beta_{M,0}$ to the second summation

$$= \frac{M-1}{2} \mu A_{M,0} \quad \square$$

Lemma 2.3. For any method written in the form (2.1) above,

$$A_{M,k} \leq \frac{M-k}{k+1} \mu A_{M,k-1}$$

for any $1 \leq M \leq m$, where $\mu = \max_{i,k} \frac{\beta_{i,k}}{\alpha_{i,k}}$.

Proof. Using Lemma 2.2 as the base case, we show that if

$$A_{p,l} \leq \frac{p-l}{l+1} \mu A_{p,l-1} \quad \text{for } 1 \leq p \leq M-1 \quad \text{and} \quad 1 \leq l \leq p-1$$

then for any $k < M-1$,

$$\begin{aligned} A_{M,k} &= \sum_{j=k+1}^{M-1} \alpha_{M,j} A_{j,k} + \sum_{j=k}^{M-1} \beta_{M,j} A_{j,k-1} \\ &\leq \sum_{j=k+1}^{M-1} \alpha_{M,j} \frac{j-k}{k+1} \mu A_{j,k-1} + \sum_{j=k}^{M-1} \beta_{M,j} A_{j,k-1} \\ &= \sum_{j=k}^{M-1} \alpha_{M,j} \frac{j-k}{k+1} \mu A_{j,k-1} + \sum_{j=k}^{M-1} \beta_{M,j} A_{j,k-1} \end{aligned}$$

where the zero term is added to the first summation

$$\begin{aligned} &\leq \frac{M-1-k}{k+1} \mu \sum_{j=k}^{M-1} \alpha_{M,j} A_{j,k-1} + \sum_{j=k}^{M-1} \beta_{M,j} A_{j,k-1} \\ &= \frac{M-k}{k+1} \mu \sum_{j=k}^{M-1} \alpha_{M,j} A_{j,k-1} + \sum_{j=k}^{M-1} \left(\beta_{M,j} - \frac{1}{k+1} \mu \alpha_{M,j} \right) A_{j,k-1} \\ &= \frac{M-k}{k+1} \mu \sum_{j=k}^{M-1} \alpha_{M,j} A_{j,k-1} + \sum_{j=k}^{M-1} \left(\mu_{M,j} - \frac{1}{k+1} \mu \right) \alpha_{M,j} A_{j,k-1} \\ &\leq \frac{M-k}{k+1} \mu \sum_{j=k}^{M-1} \alpha_{M,j} A_{j,k-1} + \sum_{j=k}^{M-1} \left(\mu_{M,j} - \frac{1}{k+1} \mu_{M,j} \right) \alpha_{M,j} A_{j,k-1} \end{aligned}$$

because $\mu_{M,j} \leq \mu$

$$\begin{aligned} &= \frac{M-k}{k+1} \mu \sum_{j=k}^{M-1} \alpha_{M,j} A_{j,k-1} + \sum_{j=k}^{M-1} \left(1 - \frac{1}{k+1} \right) \beta_{M,j} A_{j,k-1} \\ &= \frac{M-k}{k+1} \mu \sum_{j=k}^{M-1} \alpha_{M,j} A_{j,k-1} + \sum_{j=k}^{M-1} \frac{k}{k+1} \beta_{M,j} A_{j,k-1} \\ &\leq \frac{M-k}{k+1} \mu \sum_{j=k}^{M-1} \alpha_{M,j} A_{j,k-1} + \sum_{j=k}^{M-1} \frac{k}{k+1} \beta_{M,j} \frac{j-k+1}{k} \mu A_{j,k-2} \\ &\leq \frac{M-k}{k+1} \mu \sum_{j=k}^{M-1} \alpha_{M,j} A_{j,k-1} + \frac{k}{k+1} \frac{M-k}{k} \mu \sum_{j=k-1}^{M-1} \beta_{M,j} A_{j,k-2} \end{aligned}$$

by adding the nonnegative term $\beta_{M,k-1} A_{k-1,k-2}$ to the second summation

$$\begin{aligned}
&= \frac{M-k}{k+1} \mu \left(\sum_{j=k}^{M-1} \alpha_{M,j} A_{j,k-1} + \sum_{j=k-1}^{M-1} \beta_{M,j} A_{j,k-2} \right) \\
&= \frac{M-k}{k+1} \mu A_{M,k-1}
\end{aligned}$$

Finally, for the case $k = M - 1$,

$$\begin{aligned}
A_{M,M-1} &= \beta_{M,M-1} A_{M-1,M-2} \\
&= \mu_{M,M-1} \alpha_{M,M-1} A_{M-1,M-2} \\
&= \frac{1}{M} \mu \alpha_{M,M-1} A_{M-1,M-2} - \frac{1}{M} \mu \alpha_{M,M-1} A_{M-1,M-2} \\
&\quad + \mu_{M,M-1} \alpha_{M,M-1} A_{M-1,M-2} \\
&= \frac{1}{M} \mu \alpha_{M,M-1} A_{M-1,M-2} + \left(\mu_{M,M-1} - \frac{1}{M} \mu \right) \alpha_{M,M-1} A_{M-1,M-2} \\
&\leq \frac{1}{M} \mu \alpha_{M,M-1} A_{M-1,M-2} + \left(\mu_{M,M-1} - \frac{1}{M} \mu_{M,M-1} \right) \alpha_{M,M-1} A_{M-1,M-2} \\
&= \frac{1}{M} \mu \alpha_{M,M-1} A_{M-1,M-2} + \frac{M-1}{M} \mu_{M,M-1} \alpha_{M,M-1} A_{M-1,M-2} \\
&= \frac{1}{M} \mu \alpha_{M,M-1} A_{M-1,M-2} + \frac{M-1}{M} \beta_{M,M-1} A_{M-1,M-2} \\
&\leq \frac{1}{M} \mu \alpha_{M,M-1} A_{M-1,M-2} + \frac{M-1}{M} \beta_{M,M-1} \frac{M-1-M+2}{M-1} \mu A_{M-1,M-3} \\
&= \frac{1}{M} \mu \alpha_{M,M-1} A_{M-1,M-2} + \frac{1}{M} \mu \beta_{M,M-1} A_{M-1,M-3} \\
&\leq \frac{1}{M} \mu \alpha_{M,M-1} A_{M-1,M-2} + \frac{1}{M} \mu \beta_{M,M-1} A_{M-1,M-3} \\
&\quad + \frac{1}{M} \mu \beta_{M,M-2} A_{M-2,M-3} \\
&= \frac{1}{M} \mu A_{M,M-2}
\end{aligned}$$

□

2.2. Upper Bound for the Optimal CFL Number of a m Stage N th Order Method

The lemmas in the preceding section suggest a bound on the optimal size of the CFL number c , which depends on the number of stages m and the order N of the method.

Proposition 2.1. Consider the family of m -stage, N th order SSP Runge–Kutta methods (1.7) with nonnegative coefficients $\alpha_{i,k}$ and $\beta_{i,k}$. The CFL number c in (1.5) will be, at most, $c = m - N + 1$.

Proof. From the lemmas above, we see that for any $M \geq 1$

$$A_{M,k} \leq \frac{M-k}{k+1} \mu A_{M,k-1} \quad \text{for } 1 \leq k < M$$

and

$$A_{M,0} \leq M\mu$$

For an m -stage method to be N th order, we must have [5]

$$A_{m,n} = \frac{1}{(n+1)!} \quad \text{for } n = 0, 1, \dots, N-1 \quad (2.2)$$

so we have

$$\frac{1}{N!} = A_{m,N-1} \leq \frac{m-N+1}{N} \mu A_{m,N-2} = \frac{m+1-N}{N} \mu \frac{1}{(N-1)!}$$

Consequently,

$$\frac{1}{m+1-N} \leq \mu$$

The CFL number c would be, at most, $c = m + 1 - N$. □

This, however, is only a bound, and does not mean that such a CFL number can actually be obtained. As we will show, there are many cases in which this optimal CFL number is indeed attainable.

2.3. Some Optimal SSP Runge–Kutta Methods

In this section we construct some m -stage N th order SSP methods with optimal CFL. We start with a first order, m stage method with CFL $c = m$:

Proposition 2.2. The m stage method given by

$$\begin{aligned} u^{(0)} &= u^n \\ u^{(i)} &= \left(1 + \frac{\Delta t}{m} L\right) u^{(i-1)}, \quad i = 1, \dots, m \\ u^{n+1} &= u^{(m)} \end{aligned} \quad (2.3)$$

is first order accurate, with CFL number $c = m$.

Proof. Since for each nonzero $\alpha_{i,k}$ we have $\alpha_{i,k} = 1$ and $\beta_{i,k} = \frac{1}{m}$, and $\beta_{i,k} = 0$ whenever $\alpha_{i,k} = 0$, we have $\beta_{i,k} = \frac{1}{m} \alpha_{i,k}$ and the CFL number $c = m$ is clear. To check that the order, we notice that

$$\begin{aligned} u^{n+1} &= \left(1 + \Delta t \frac{1}{m} L\right)^m u^n \\ &= \left(1 + \Delta t L + \Delta t^2 \frac{m-1}{2m} L^2 + \dots + \frac{1}{m^m} \Delta t^m L^m\right) u^n \end{aligned}$$

since $\frac{m-1}{2m} \neq \frac{1}{2}$ for any m , we have

$$u^{n+1} = (1 + \Delta t L + O(\Delta t^2)) u^n$$

a first order method. □

We note that although this method has a significantly higher CFL than the standard first order method (which is, of course, the forward-Euler method), it has a correspondingly higher computational cost. Although the stepsize can be increased by a factor of m , the computational cost is also increased by the same factor. We need to look not only at the CFL number, but also at the number of steps needed. To reflect this, we define the *effective* CFL number $c_{\text{eff}} = s_r c$ where s_r is the ratio of the number of steps needed for the standard method to the number of steps needed for the current method. Thus, for the method (2.2) the effective CFL is, in fact, $c_{\text{eff}} = 1$. However, this method is useful as a stepping-stone for higher order methods.

For any desired integer optimal CFL number c , a first order ($N = 1$) method of this CFL number is then given (as in Proposition 2.2 above) by the m -stage method:

$$\begin{aligned} u^{(0)} &= u^n \\ u^{(i)} &= (1 + \Delta t \mu L) u^{(i-1)}, \quad i = 1, \dots, m \\ u^{n+1} &= u^{(m)} \end{aligned} \quad (2.4)$$

where $m = c$ and $\mu = \frac{1}{c}$. The next proposition shows how we can recursively build higher order methods. Starting with this method as a building block, we add one stage and increase the order to two, without changing the CFL number. Following the procedure detailed below, we can then build m stage, $N = m + 1 - c$ order methods with the optimal CFL number c chosen above. However, there is no guarantee that these methods will prove to be SSP.

Proposition 2.3. For any given CFL number $c = \frac{1}{\mu}$, where μ is chosen so that c is a positive integer, the class of m stage, $N = (m + 1 - c)$ order schemes of the form

$$\begin{aligned} u^{(0)} &= u^n \\ u^{(i)} &= u^{(i-1)} + \mu \Delta t Lu^{(i-1)}, \quad i = 1, \dots, m-1 \\ u^{(m)} &= \sum_{k=0}^{m-2} \alpha_{m,k} u^{(k)} + \alpha_{m,m-1} (u^{(m-1)} + \mu \Delta t Lu^{(m-1)}), \\ u^{n+1} &= u^{(m)} \end{aligned} \quad (2.5)$$

is given recursively by the coefficients:

$$\begin{aligned} \alpha_{m,k} &= \frac{1}{k\mu} \alpha_{m-1,k-1}, \quad k = 1, \dots, m-2 \\ \alpha_{m,m-1} &= \frac{1}{m\mu} \alpha_{m-1,m-2}, \quad \alpha_{m,0} = 1 - \sum_{k=1}^{m-1} \alpha_{m,k} \end{aligned} \quad (2.6)$$

where the initial method is that given by the c -stage, first order method (2.4) above.

Proof. In (2.5), for each $1 \leq i \leq m-1$

$$\begin{aligned} u^{(i)} &= u^{(i-1)} + \mu \Delta t Lu^{(i-1)} \\ &= (1 + \mu \Delta t L)^i u^{(0)} \end{aligned}$$

We rewrite the method (2.5) above as

$$\begin{aligned}
u^{(n+1)} &= \sum_{k=0}^{m-2} \alpha_{m,k} (1 + \mu \Delta t L)^k u^{(0)} + \alpha_{m,m-1} (1 + \mu \Delta t L)(1 + \mu \Delta t L)^{m-1} u^{(0)} \\
&= \left(\sum_{k=0}^{m-2} \alpha_{m,k} \sum_{j=0}^k \frac{k!}{j!(k-j)!} \mu^j \Delta t^j L^j + \alpha_{m,m-1} \sum_{j=0}^m \frac{m!}{j!(m-j)!} \mu^j \Delta t^j L^j \right) u^{(0)} \\
&= \left(\sum_{j=0}^m \alpha_{m,j} + \left(\sum_{j=1}^{m-2} \alpha_{m,j} \frac{j!}{(j-1)!} + \alpha_{m,m-1} \frac{m!}{(m-1)!} \right) \mu \Delta t L \right. \\
&\quad + \left(\sum_{j=2}^{m-2} \alpha_{m,j} \frac{j!}{2!(j-2)!} + \alpha_{m,m-1} \frac{m!}{2!(m-2)!} \right) \mu^2 \Delta t^2 L^2 \\
&\quad + \left(\sum_{j=3}^{m-2} \alpha_{m,j} \frac{j!}{3!(j-3)!} + \alpha_{m,m-1} \frac{m!}{3!(m-3)!} \right) \mu^3 \Delta t^3 L^3 + \dots \\
&\quad \left. + \left(\sum_{j=m-2}^{m-2} \alpha_{m,m-2} \frac{j!}{(j-m+2)!(m-2)!} + \alpha_{m,m-1} \frac{m!}{(m-2)!2!} \right) \mu^{m-2} \Delta t^{m-2} L^{m-2} \right. \\
&\quad \left. + \alpha_{m,m-1} \frac{m!}{(m-1)!} \mu^{m-1} \Delta t^{m-1} L^{m-1} + \alpha_{m,m-1} \mu^m \Delta t^m L^m \right) u^{(0)}
\end{aligned}$$

For this method to be N th order, we must match this with the desired expansion

$$u^{(n+1)} = \left(1 + \Delta t L + \frac{1}{2} \Delta t^2 L^2 + \frac{1}{3!} \Delta t^3 L^3 + \dots + \frac{1}{N!} \Delta t^N L^N + O(\Delta t^{N+1}) \right) u^{(0)}$$

Clearly, for the m -stage method of the type (1.3) to be N th order, the coefficients $\alpha_{i,k}$ must satisfy the order conditions:

$$(\mu)^k \left(\sum_{j=k}^{m-2} \frac{j!}{(j-k)!} \alpha_{m,j} + \frac{(m)!}{(m-k)!} \alpha_{m,m-1} \right) = 1$$

for $k = 0, \dots, N$. Correspondingly, the coefficients of a $(m+1)$ -stage, $(N+1)$ order method must satisfy

$$(\mu)^k \left(\sum_{j=k}^{m-1} \frac{j!}{(j-k)!} \alpha_{m+1,j} + \frac{(m+1)!}{(m+1-k)!} \alpha_{m+1,m} \right) = 1$$

for $k = 0, \dots, N+1$.

Assume that we have a m stage N order method of the type (2.5). Using the recursive definition we obtain the coefficients of a $(m+1)$ stage method of the same type. The definition of $\alpha_{m+1,0}$ guarantees the correct $k=0$ order condition for the $(m+1)$ stage method. We proceed to show that the k th order condition for the m stage method together with the definition of the coefficients implies the $k+1$ order condition for the $m+1$ stage method:

$$\begin{aligned} 1 &= (\mu)^k \left(\sum_{j=k}^{m-2} \frac{j!}{(j-k)!} \alpha_{m,j} + \frac{(m)!}{(m-k)!} \alpha_{m,m-1} \right) \\ &= (\mu)^k \left(\sum_{j=k}^{m-2} \frac{j!}{(j-k)!} (j+1) \mu \alpha_{m+1,j+1} + \frac{(m)!}{(m-k)!} (m+1) \mu \alpha_{m+1,m} \right) \\ &= (\mu)^{k+1} \left(\sum_{j=k+1}^{m-1} \frac{j!}{(j-(k+1))!} \alpha_{m+1,j} + \frac{(m+1)!}{((m+1)-(k+1))!} \alpha_{m+1,m} \right) \end{aligned}$$

The $k=0, \dots, N$ order conditions for the m stage method imply the $k=1, \dots, N+1$ order conditions for the $(m+1)$ stage method, and the $k=0$ order condition is true by definition. Thus, the order conditions for $k=0, \dots, N+1$ are satisfied and the $(m+1)$ stage method will be of order $(N+1)$. \square

A scheme obtained in this way is SSP with CFL $c = \frac{1}{\mu}$ as long as the coefficients $\alpha_{i,k}$ are nonnegative. However, not all the methods generated in this way are SSP—most of them will have negative $\alpha_{i,k}$. Nevertheless, this method is useful for generating the following methods:

Method 1. The following are second order ($N=2$) SSP methods with m stages and an optimal CFL number $c = m-1$:

$$\begin{aligned} u^{(0)} &= u^n \\ u^{(i)} &= \left(1 + \frac{\Delta t}{m-1} L \right) u^{(i-1)}, \quad i = 1, \dots, m-1 \\ u^m &= \frac{1}{m} u^{(0)} + \frac{m-1}{m} \left(1 + \frac{\Delta t}{m-1} L \right) u^{(m-1)} \\ u^{n+1} &= u^{(m)} \end{aligned} \tag{2.7}$$

The CFL number of this method is clear by inspection. A quick verification of the order of this scheme follows:

$$\begin{aligned}
u^{n+1} &= \frac{1}{m} u^{(0)} + \frac{m-1}{m} \left(1 + \frac{\Delta t}{m-1} L \right)^m u^{(0)} \\
&= \left(\frac{1}{m} + \frac{m-1}{m} \left(1 + m \frac{\Delta t}{m-1} L + \frac{m(m-1)}{2} \frac{\Delta t^2}{(m-1)^2} L^2 + O(\Delta t^3) \right) \right) u^{(0)} \\
&= \left(\frac{1}{m} + \frac{m-1}{m} + \Delta t L + \frac{1}{2} \Delta t^2 L^2 + O(\Delta t^3) \right) u^{(0)} \\
&= \left(1 + \Delta t L + \frac{1}{2} \Delta t^2 L^2 + O(\Delta t^3) \right) u^{(0)}
\end{aligned}$$

In fact, these methods are also nonlinearly second order [16]. Each such method uses m stages to attain the order usually obtained by a 2-stage method, but has CFL number $m-1$, thus the effective CFL number here is increased to $c_{\text{eff}} = \frac{2(m-1)}{m}$.

Method 2. Using the method in proposition (2.3) we generate methods of any order N with $m = N + 1$ stages, which are SSP with CFL coefficient $c = 2$. Table I includes the coefficients of these methods. The effective CFL for these methods is also $c_{\text{eff}} = \frac{2N}{N+1} = \frac{2(m-1)}{m}$.

3. LINEAR CONSTANT COEFFICIENT OPERATORS WITH TIME DEPENDENT FORCING TERMS

As we have seen [5], SSP Runge Kutta methods suitable for a linear, constant coefficient ODE are easier to generate and have a higher CFL

Table I. Coefficients $\alpha_{m,j}$ of the m -Stage $N = (m-1)$ Order SSP Methods of the Form (2.5), Which Have CFL Number $c = 2$

stages m	$\alpha_{m,0}$	$\alpha_{m,1}$	$\alpha_{m,2}$	$\alpha_{m,3}$	$\alpha_{m,4}$	$\alpha_{m,5}$	$\alpha_{m,6}$	$\alpha_{m,7}$	$\alpha_{m,8}$	$\alpha_{m,9}$
2	0	1								
3	$\frac{1}{3}$	0	$\frac{2}{3}$							
4	0	$\frac{2}{3}$	0	$\frac{1}{3}$						
5	$\frac{1}{5}$	0	$\frac{2}{3}$	0	$\frac{2}{15}$					
6	$\frac{1}{9}$	$\frac{2}{5}$	0	$\frac{4}{9}$	0	$\frac{2}{45}$				
7	$\frac{1}{7}$	$\frac{2}{9}$	$\frac{2}{5}$	0	$\frac{2}{9}$	0	$\frac{4}{315}$			
8	$\frac{2}{15}$	$\frac{2}{7}$	$\frac{2}{9}$	$\frac{4}{15}$	0	$\frac{4}{45}$	0	$\frac{1}{315}$		
9	$\frac{11}{81}$	$\frac{4}{15}$	$\frac{2}{7}$	$\frac{4}{27}$	$\frac{2}{15}$	0	$\frac{4}{135}$	0	$\frac{2}{2835}$	
10	$\frac{71}{525}$	$\frac{22}{81}$	$\frac{4}{15}$	$\frac{4}{21}$	$\frac{2}{27}$	$\frac{4}{75}$	0	$\frac{8}{945}$	0	$\frac{2}{14175}$

than SSP Runge Kutta methods for a nonlinear ODE. We wish to extend these nice results to the case of a constant linear operator with a time dependent forcing term. This is a case which also arises in linear PDEs with time dependent boundary conditions, and can be written as:

$$u_t = Lu + f(t) \quad (3.1)$$

where $u = [u_i]$ is a vector, $L = [L_{i,j}]$ is a constant matrix and $f(t) = [f_i(t)]$ is a vector of functions of t . This ODE is a linear time dependent ODE and as such, the Runge–Kutta methods derived above for a linear time-invariant ODE will not have the correct order. The problem is that the class of RK methods for linear, time dependent ODEs is not equivalent to those for linear time invariant ODEs [20]. However, if the functions $f(t)$ can be written in a suitable way, then we can convert the equation (3.1) to a linear constant-coefficient ODE.

The order conditions for a RK method are derived, without loss of generality [2], for autonomous system $y'(x) = g(y(x))$. The reason for the “no loss of generality” is that any system of the form

$$u'(x) = h(x, u(x))$$

can be converted to an autonomous system by setting

$$y(x) = \begin{pmatrix} x \\ u(x) \end{pmatrix}$$

and then

$$y'(x) = g(y(x)) = g \begin{pmatrix} x \\ u(x) \end{pmatrix} = \begin{pmatrix} 1 \\ h(x, u) \end{pmatrix}$$

In many cases, we can convert equation (3.1) to a linear, constant coefficient ODE using a similar transformation. We first write (or approximate, if necessary) $f(t)$ as

$$f_i(t) = \sum_{j=0}^n a_j^i q_j(t) = Aq(t)$$

where $A = [A_{i,j}] = [a_j^i]$ is a constant matrix and $q(t) = [q_j(t)]$ are a set of functions which have the property that $q'(t) = Dq(t)$, where D is a constant matrix. Once the approximation to $f(t)$ is obtained, the ODE (3.1) can be converted into the linear, constant coefficient ODE

$$y_t = My(t) \quad (3.2)$$

where

$$y(t) = \begin{pmatrix} q(t) \\ u(t) \end{pmatrix}$$

and

$$M = \begin{pmatrix} D & 0 \\ A & L \end{pmatrix}$$

Thus, an equation of the form (3.1) can be approximated (or given exactly) by an linear constant coefficient ODE, and the SSP Runge–Kutta methods derived in Sec. 2.3 can be applied to this case.

Remark. We stress that we are talking about preserving the stability properties of forward-Euler as applied to the equation $y_t = My$. It is possible (indeed, expected) that forward-Euler applied to $u_t = Lu$ may satisfy properties not satisfied when applied to $u_t = Lu + f(t)$. It is also possible that some properties satisfied by forward-Euler when applied to the exact equation $u_t = Lu + f(t)$ may not be satisfied once $f(t)$ is approximated.

Remark. To approximate the functions $f(t)$ we can use the polynomials $q_j(t) = t^j$. In this case, the differentiation matrix D is given by

$$D = \begin{pmatrix} 0 & 0 & 0 & \cdots & 0 & 0 \\ 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 2 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 3 & \cdots & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 0 & \cdots & n & 0 \end{pmatrix}$$

A better approximation can be obtained using the Chebyshev polynomials. For these polynomials the relationship between the polynomial and its derivative is given by $T'_n(x) = \sum_{j=0}^n b_j T_j(x)$ where (see [6]),

$$b_j = \begin{cases} n & \text{for } j=0, \text{ if } n \text{ is odd} \\ 2n & \text{for } j>0, \text{ if } j+n \text{ is odd} \end{cases}$$

In other words, the derivative matrix D takes the form:

$$D = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 3 & 0 & 6 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 8 & 0 & 8 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 5 & 0 & 10 & 0 & 10 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 12 & 0 & 12 & 0 & 12 & 0 & 0 & \cdots & 0 & 0 \\ 7 & 0 & 14 & 0 & 14 & 0 & 14 & 0 & \cdots & 0 & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdots & \cdot & \cdot \\ n & 0 & 2n & 0 & 2n & 0 & 2n & 0 & \cdots & 2n & 0 \end{pmatrix} \quad \text{if } n \text{ is odd}$$

or

$$D = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 3 & 0 & 6 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 8 & 0 & 8 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 5 & 0 & 10 & 0 & 10 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 12 & 0 & 12 & 0 & 12 & 0 & 0 & \cdots & 0 & 0 \\ 7 & 0 & 14 & 0 & 14 & 0 & 14 & 0 & \cdots & 0 & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdots & \cdot & \cdot \\ 0 & 2n & 0 & 2n & 0 & 2n & 0 & 2n & \cdots & 2n & 0 \end{pmatrix} \quad \text{if } n \text{ is even}$$

4. NUMERICAL RESULTS

We approximate the solution to the equation

$$u_t = u_{xx} + 4t^3 \quad 0 \leq x \leq \pi \quad (4.1)$$

with initial condition

$$u(x, 0) = \sin(x)$$

and boundary conditions

$$u(0, t) = u(\pi, t) = t^4$$

This equation has the exact solution

$$u(x, t) = t^4 + e^{-t} \sin(x)$$

We employ the second order centered difference spatial discretization

$$u_{xx} \approx \frac{u_{j+1} - 2u_j + u_{j-1}}{\Delta x^2}$$

which gives us the linear operator

$$L = \frac{1}{\Delta x^2} \begin{pmatrix} -2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & -2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -2 & 1 & 0 & 0 & 0 & 0 & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -2 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -2 \end{pmatrix}$$

in

$$u_t = Lu + 4t^3 \tag{4.2}$$

To incorporate the time-dependent boundary conditions as well as the time dependent forcing term, we define

$$y = \begin{pmatrix} 1 \\ t \\ t^2 \\ t^3 \\ t^4 \\ u \end{pmatrix}$$

and the ODE becomes

$$y_t = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 0 & 4 & \frac{1}{\Delta x^2} & \frac{-2}{\Delta x^2} & \frac{1}{\Delta x^2} & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 0 & 4 & 0 & \frac{1}{\Delta x^2} & \frac{-2}{\Delta x^2} & \frac{1}{\Delta x^2} & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 0 & 4 & 0 & 0 & \frac{1}{\Delta x^2} & \frac{-2}{\Delta x^2} & \frac{1}{\Delta x^2} & 0 & \cdots & 0 & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & 4 & 0 & 0 & \cdots & 0 & 0 & 0 & \frac{1}{\Delta x^2} & \frac{-2}{\Delta x^2} & \frac{1}{\Delta x^2} \\ 0 & 0 & 0 & 4 & \frac{1}{\Delta x^2} & 0 & \cdots & 0 & 0 & 0 & 0 & \frac{1}{\Delta x^2} & \frac{-2}{\Delta x^2} \end{pmatrix} y$$

In all these numerical experiments we use $\Delta x = \frac{1}{101}$. The following time discretizations were used:

1. The first order forward-Euler discretization:

$$y^{n+1} = (1 + \Delta t L) y^n$$

2. The 6-stage ($m = 6$), 5th order ($N = 5$) method with CFL number $c = 2$, given in Table I:

$$u^{(0)} = u^n$$

$$u^{(i)} = \left(1 + \frac{\Delta t}{2} L\right) u^{(i-1)}, \quad i = 1, \dots, 5$$

$$u^{(6)} = \frac{1}{9} u^{(0)} + \frac{2}{5} u^{(1)} + \frac{4}{9} u^{(3)} + \frac{2}{45} \left(1 + \frac{\Delta t}{2} L\right) u^{(5)}$$

$$u^{n+1} = u^{(6)}$$

The high order Runge–Kutta method was compared to the forward-Euler method. As predicted, the maximal time-step Δt allowed was doubled for

method (2) compared to the forward-Euler method. In Figs. 1 and 2 we see the effects of instability in the forward-Euler method when the time-step Δt is too high. This method is stable when $\Delta t \leq \frac{1}{2} \Delta x^2 = \Delta t_1$, however once Δt is increased to $\Delta t = 0.51 \Delta x^2$, the method becomes unstable in 400 iterations (Fig. 1). If we increase Δt to $\Delta t = \frac{3}{4} \Delta x^2$, the method becomes unstable in 20 iterations, and when $\Delta t = \Delta x^2$, the instability has destroyed the solution completely by 15 iterations (Fig. 2). The 6-stage, 5th order method is stable as long as $\Delta t \leq \Delta x^2$. Figure 3 shows that for $\Delta t = \Delta x^2$, the method is stable even at final time $t = 0.010195$, however, when the time step is raised to $\Delta t = 1.15 \Delta x^2$, the wiggles characteristic of instability are apparent at time $t = 0.010146$, or 90 iterations. As expected, the time step allowed doubled.

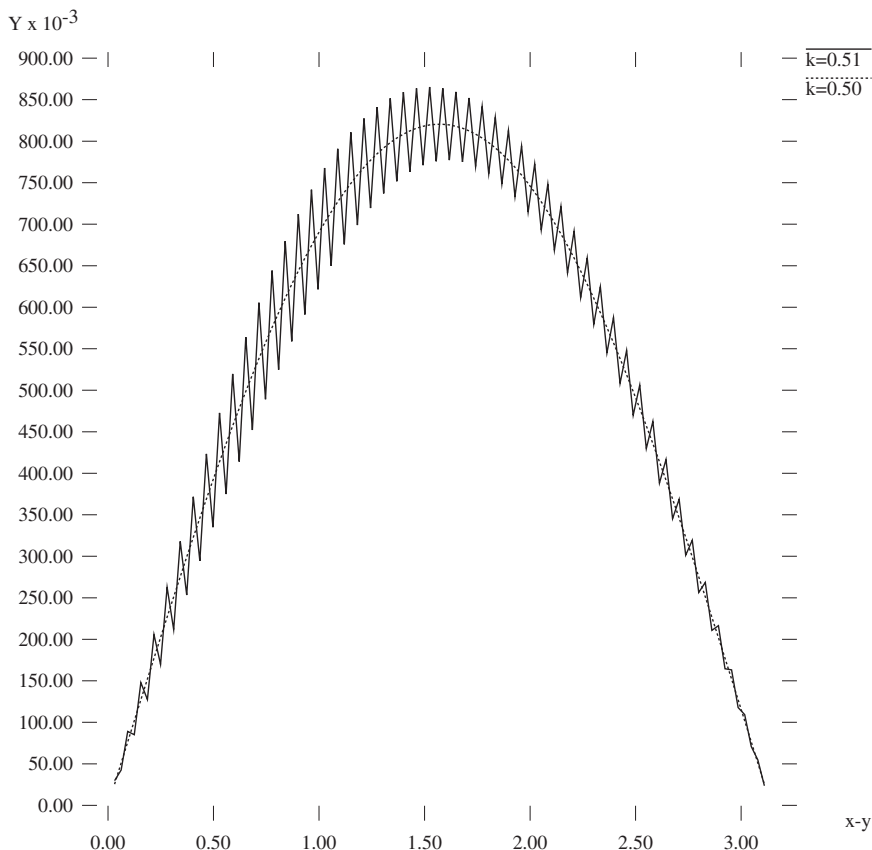


Fig. 1. Forward-Euler applied to the test problem. In each case, $\Delta t = k \Delta x^2$, where $\Delta x = \frac{1}{101}$. When $k \leq 0.5$ the method is stable. The numerical solution is shown for $k = 0.5$ after 408 time steps and for $k = 0.51$ after 400 time steps (final time = 0.020047).

numerical solution

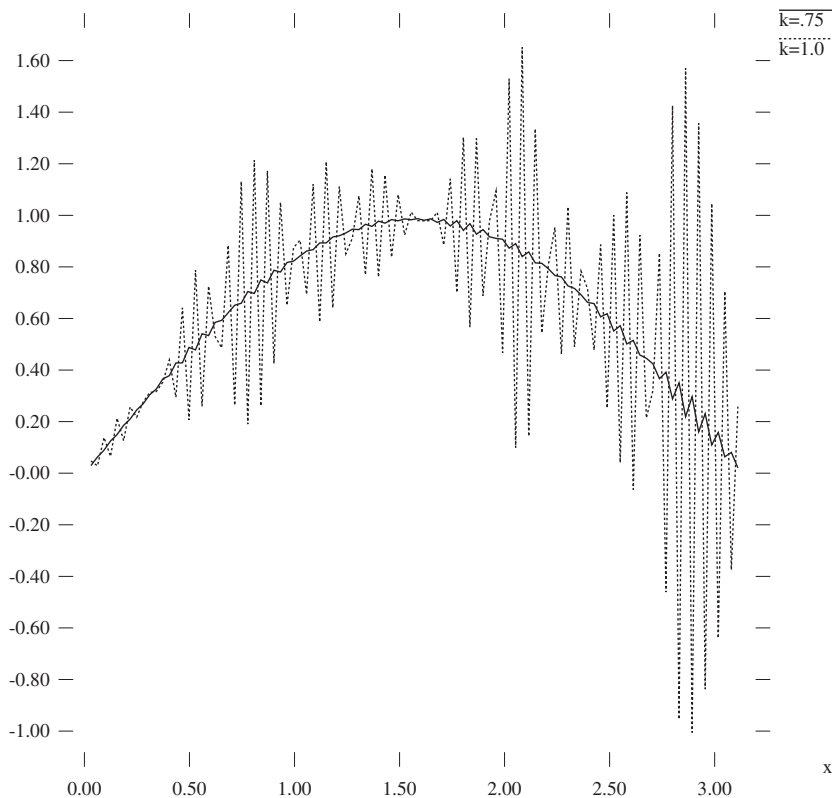


Fig. 2. Forward-Euler applied to the test problem. Once again, $\Delta t = k \Delta x^2$, where $\Delta x = \frac{1}{101}$. The method becomes unstable after very few time steps. Results shown are for $k = 0.75$ after 20 time steps and for $k = 1.0$ after 15 time steps. The instability apparent in the $k = 0.75$ case is worsened in the $k = 1.0$ case. We notice that the extent of instability in this example, for a fixed final time, depends not on the number of time steps, but mainly on the size of Δt .

An interesting point which arised from the technique used in Sec. 3 is that the time accuracy of the method is now important from the point of view of the first few elements in the new vector y . Since the time dependent boundary conditions or forcing is now not given explicitly, but by its differential equation $q_t = Dq$, the time-stepping method must also solve this ODE. If the time stepping method is not of a high enough order, the boundary conditions or forcing may not be resolved properly. In Fig. 4 we see the effect of numerically solving the ODE above on the term t^4 . A method of fourth order or above will solve this exactly. We see that the

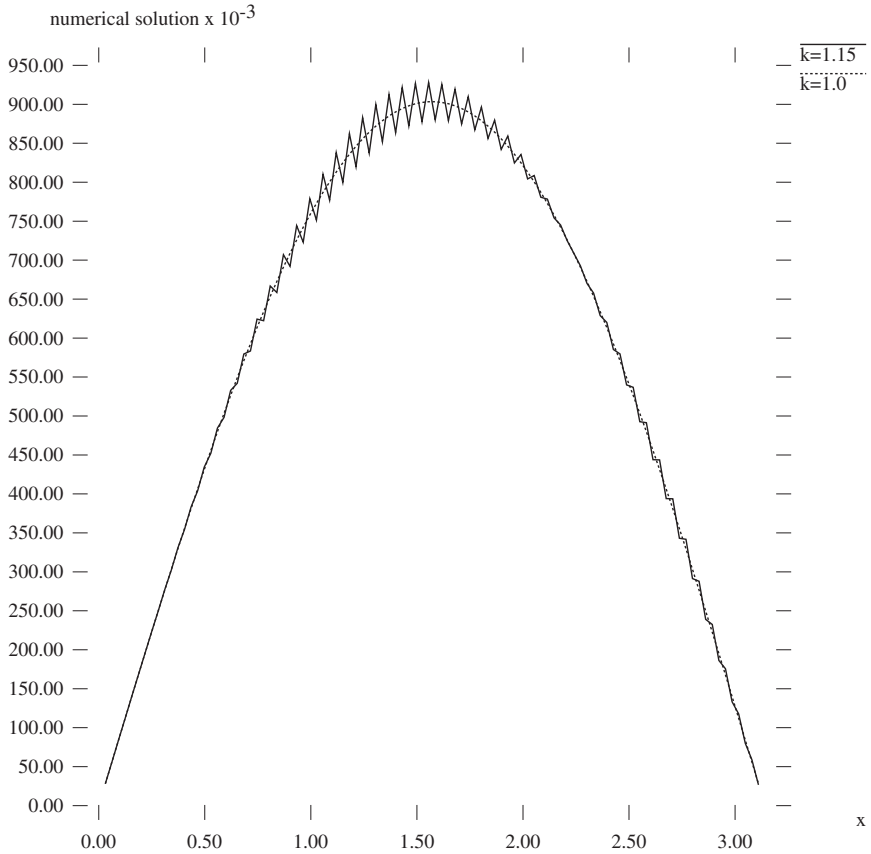


Fig. 3. The 5th order, 6 stage SSP Runge–Kutta method with CFL number $c = 2$ is applied to the test problem. Once again, $\Delta t = k \Delta x^2$, where $\Delta x = \frac{1}{101}$. The CFL of the SSP method guarantees that this method will be stable with double the maximal allowed step-size for the forward-Euler method. These results illustrate that the method is stable for $\Delta t \leq \Delta x^2$, and becomes unstable shortly after. The two curves shown are the numerical solutions for $k = 1.00$ after 104 time steps (final time=0.010195) and for $k = 1.15$ after 90 time steps (final time=0.010146). The instability is apparent in the $k = 1.15$ case.

fifth order method (2) solves it exactly, but a first order method (1) does not.

5. CONCLUDING REMARKS

While the development of SSP Runge–Kutta methods was primarily geared toward nonlinear operators, the wide applicability of these methods

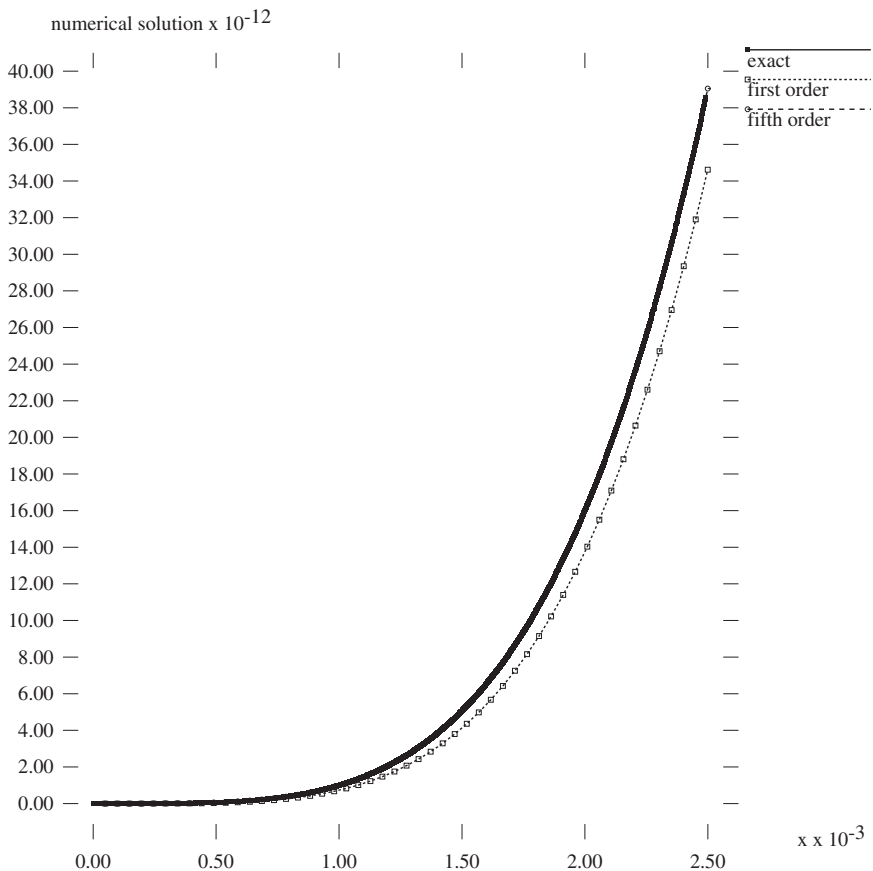


Fig. 4. The technique of Sec. 3 involves rewriting the forcing term $4t^3$ and the boundary conditions t^4 as the differential equation that governs them. This figure shows how t^4 is approximated by the fifth order method and the first order forward-Euler. As expected, the fifth order method captures the curve exactly while the first order method does not.

have motivated us to consider SSP methods for linear, time invariant operators. In [5] we presented a class of linear SSP Runge Kutta methods with m stages and of order m , which had optimal CFL number $c = 1$. Here we present a class of first order m stage methods with CFL $c = m$, a class of second order m stage methods with CFL $c = m - 1$ and a class of $m - 1$ order, m stage methods with CFL $c = 2$. We show that these methods are optimal, and that the optimal CFL for a N th order m stage method is, at most, $c = m - N + 1$. Although these results are not, in general, extendable to ODEs with time-dependent linear operators, we extend it to a special

case of this class, which proves useful for linear PDEs with time dependent forcing or boundary conditions.

ACKNOWLEDGMENTS

Research supported by NSF Grant DMS-0106743.

REFERENCES

1. Allen, M. B., and Isaacson, E. L. (1998). *Numerical Analysis for Applied Science*, John Wiley, New York.
2. Butcher, J. C. (1987). *The Numerical Analysis of Ordinary Differential Equations: Runge-Kutta and General Linear Methods*, John Wiley, New York,.
3. Cockburn, B., and Shu, C.-W. (1989). TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws II: General framework. *Math. Comp.* **52**, 411-435.
4. Gottlieb, S., and Shu, C.-W. (1998). Total variation diminishing Runge-Kutta schemes. *Math. Comp.* **67**, 73-85.
5. Gottlieb, S., Shu, C.-W., and Tadmor, E. (2001). Strong stability preserving high order time discretization methods. *SIAM Rev.* **43**(1), 89-112.
6. Gottlieb, D., and Orszag, S. A. (1977). *Numerical Analysis of Spectral Methods: Theory and Applications*, Society for Industrial and Applied Mathematics, Philadelphia.
7. Gottlieb, D., and Tadmor, E. (1991). The CFL condition for spectral approximations to hyperbolic initial-boundary value problems. *Math. Comp.* **56**, 565-588.
8. Gustafsson, B., Kreiss, H. O., and Olinger, J. (1995). *Time Dependent Problems and Difference Methods*, John Wiley, New York.
9. Harten, A. (1983). High resolution schemes for hyperbolic conservation laws. *J. Comput. Phys.* **49**, 357-393.
10. Kurganov, A., and Tadmor, E. (1999). New high-resolution schemes for nonlinear conservation laws and related convection-diffusion equations, UCLA CAM Report No. 99-16.
11. Levy, D., and Tadmor, E. (1998). From semi-discrete to fully discrete: Stability of Runge-Kutta schemes by the energy method. *SIAM Rev.* **40**, 40-73.
12. Osher, S., and Chakravarthy, S. (1984). High resolution schemes and the entropy condition. *SIAM J. Numer. Anal.* **21**, 955-984.
13. Osher, S., and Tadmor, E. (1988). On the convergence of difference approximations to scalar conservation laws. *Math. Comp.* **50**, 19-51.
14. Shu, C.-W. (1988). Total-variation-diminishing time discretizations. *SIAM J. Sci. Stat. Comput.* **9**, 1073-1084.
15. Shu, C.-W., and Osher, S. (1988). Efficient implementation of essentially non-oscillatory shock-capturing schemes. *J. Comput. Phys.* **77**, 439-471.
16. Ruuth, S. J., and Spiteri, R. J. (2001). *A New Class of Optimal High-Order Strong-Stability-Preserving Methods*, Unpublished manuscript (under review).
17. Ruuth, S. J., and Spiteri, R. J. (2001). *Two Barriers on Strong Stability Preserving Time Discretization Methods*, Unpublished manuscript (under review).
18. Sweby, P. K. (1984). High resolution schemes using flux limiters for hyperbolic conservation laws. *SIAM J. Numer. Anal.* **21**, 995-1011.

19. Tadmor, E. (1997). Approximate solutions of nonlinear conservation laws. In Quarteroni, A. (ed.), *Advanced Numerical Approximation of Nonlinear Hyperbolic Equations*, Lecture Notes from CIME Course Cetraro, Italy, Lecture Notes in Mathematics 1697, Springer-Verlag, 1998, pp. 1–150.
20. Verner, J. H. (1996). High-order explicit Runge–Kutta pairs with low stage order. *Appl. Numer. Methods* **22**, 345–357.

Strong Stability-Preserving High-Order Time Discretization Methods*

Sigal Gottlieb[†]
Chi-Wang Shu[‡]
Eitan Tadmor[§]

Abstract. In this paper we review and further develop a class of strong stability-preserving (SSP) high-order time discretizations for semidiscrete method of lines approximations of partial differential equations. Previously termed TVD (total variation diminishing) time discretizations, these high-order time discretization methods preserve the strong stability properties of first-order Euler time stepping and have proved very useful, especially in solving hyperbolic partial differential equations. The new developments in this paper include the construction of optimal explicit SSP linear Runge–Kutta methods, their application to the strong stability of coercive approximations, a systematic study of explicit SSP multistep methods for nonlinear problems, and the study of the SSP property of implicit Runge–Kutta and multistep methods.

Key words. strong stability preserving, Runge–Kutta methods, multistep methods, high-order accuracy, time discretization

AMS subject classifications. 65M20, 65L06

PII. S003614450036757X

I. Introduction. It is a common practice in solving time-dependent partial differential equations (PDEs) to first discretize the spatial variables to obtain a semidiscrete method of lines scheme. This is then an ordinary differential equation (ODE) system in the time variable, which can be discretized by an ODE solver. A relevant question here concerns stability. For problems with smooth solutions, usually a linear stability analysis is adequate. For problems with discontinuous solutions, however, such as solutions to hyperbolic problems, a stronger measure of stability is usually required.

*Received by the editors February 11, 2000; accepted for publication (in revised form) August 1, 2000; published electronically February 2, 2001.

<http://www.siam.org/journals/sirev/43-1/36757.html>

[†]Department of Mathematics, University of Massachusetts at Dartmouth, Dartmouth, MA 02747 and Division of Applied Mathematics, Brown University, Providence, RI 02912 (sgottlieb@umassd.edu). The research of this author was supported by ARO grant DAAG55-97-1-0318 and NSF grant ECS-9627849.

[‡]Division of Applied Mathematics, Brown University, Providence, RI 02912 (shu@cfm.brown.edu). The research of this author was supported by ARO grants DAAG55-97-1-0318 and DAAD19-00-1-0405, NSF grants DMS-9804985, ECS-9906606, and INT-9601084, NASA Langley grant NAG-1-2070 and contract NAS1-97046 while this author was in residence at ICASE, NASA Langley Research Center, Hampton, VA 23681-2199, and by AFOSR grant F49620-99-1-0077.

[§]Department of Mathematics, University of California at Los Angeles, Los Angeles, CA 90095 (tadmor@math.ucla.edu). The research of this author was supported by NSF grant DMS97-06827 and ONR grant N00014-1-J-1076.

In this paper we review and further develop a class of high-order strong stability-preserving (SSP) time discretization methods for the semidiscrete method of lines approximations of PDEs. These time discretization methods were first developed in [20] and [19] and were called TVD (total variation diminishing) time discretizations. This class of methods was further developed in [6]. The idea is to *assume* that the first-order forward Euler time discretization of the method of lines ODE is strongly stable under a certain norm when the time step Δt is suitably restricted, and then to try to find a higher order time discretization (Runge–Kutta or multistep) that maintains strong stability for the same norm, perhaps under a different time step restriction. In [20] and [19], the relevant norm was the total variation norm: the forward Euler time discretization of the method of lines ODE was assumed to be TVD, hence the class of high-order time discretization developed there was termed *TVD time discretization*. This terminology was also used in [6]. In fact, the essence of this class of high-order time discretizations lies in its ability to maintain the strong stability in the same norm as the first-order forward Euler version, hence *SSP time discretization* is a more suitable term, which we will use in this paper.

We begin this paper by discussing explicit SSP methods. We first give, in section 2, a brief introduction to the setup and basic properties of the methods. We then move, in section 3, to our new results on optimal SSP Runge–Kutta methods of arbitrary order of accuracy for *linear* ODEs suitable for solving PDEs with linear spatial discretizations. This is used to prove strong stability for a class of well-posed problems $u_t = L(u)$, where the operator L is linear and coercive, improving and simplifying the proofs for the results in [13]. We review and further develop the results in [20], [19], and [6] for nonlinear SSP Runge–Kutta methods in section 4 and for multistep methods in section 5. Section 6 of this paper contains our new results on implicit SSP schemes. It starts with a numerical example showing the necessity of preserving the strong stability property of the method, then it moves on to the analysis of the rather disappointing negative results concerning the nonexistence of SSP implicit Runge–Kutta or multistep methods of order higher than 1. Concluding remarks are given in section 7.

2. Explicit SSP Methods.

2.1. Why SSP Methods? Explicit SSP methods were developed in [20] and [19] (termed TVD time discretizations there) to solve systems of ODEs

$$(2.1) \quad \frac{d}{dt}u = L(u),$$

resulting from a method of lines approximation of the hyperbolic conservation law,

$$(2.2) \quad u_t = -f(u)_x,$$

where the spatial derivative, $f(u)_x$, is discretized by a TVD finite difference or finite element approximation; see, e.g., [8], [16], [21], [2], [9], and consult [22] for a recent overview. Denoted by $-L(u)$, it is assumed that the spatial discretization has the property that when it is combined with the first-order forward Euler time discretization,

$$(2.3) \quad u^{n+1} = u^n + \Delta t L(u^n),$$

then, for a sufficiently small time step dictated by the Courant–Friedrichs–Levy (CFL) condition,

$$(2.4) \quad \Delta t \leq \Delta t_{FE},$$

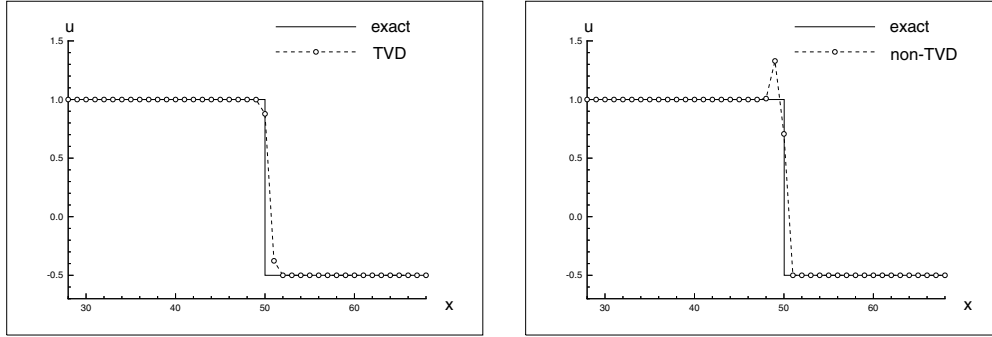


Fig. 2.1 *Second-order TVD MUSCL spatial discretization. Solution after the shock moves 50 mesh points. Left: SSP time discretization; right: non-SSP time discretization.*

the total variation (TV) of the one-dimensional discrete solution

$$u^n := \sum_j u_j^n \mathbf{1}_{\{x_{j-\frac{1}{2}} \leq x \leq x_{j+\frac{1}{2}}\}}$$

does not increase in time; i.e., the following so-called TVD property holds:

$$(2.5) \quad TV(u^{n+1}) \leq TV(u^n), \quad TV(u^n) := \sum_j |u_{j+1}^n - u_j^n|.$$

The objective of the high-order SSP Runge–Kutta or multistep time discretization is to maintain the strong stability property (2.5) while achieving higher order accuracy in time, perhaps with a modified CFL restriction (measured here with a CFL coefficient, c)

$$(2.6) \quad \Delta t \leq c \Delta t_{FE}.$$

In [6] we gave numerical evidence to show that oscillations may occur when using a linearly stable, high-order method which lacks the strong stability property, even if the same spatial discretization is TVD when combined with the first-order forward Euler time discretization. The example is illustrative, so we reproduce it here. We consider a scalar conservation law, the familiar Burgers equation

$$(2.7) \quad u_t + \left(\frac{1}{2}u^2\right)_x = 0$$

with Riemann initial data

$$(2.8) \quad u(x, 0) = \begin{cases} 1 & \text{if } x \leq 0, \\ -0.5 & \text{if } x > 0. \end{cases}$$

The spatial discretization is obtained by a second order MUSCL [12], which is TVD for forward Euler time discretization under suitable CFL restriction.

In Figure 2.1, we show the result of using an SSP second-order Runge–Kutta method for the time discretization (left) and that of using a non-SSP second-order Runge–Kutta method (right). We can clearly see that the non-SSP result is oscillatory (there is an overshoot).

This simple numerical example illustrates that it is safer to use an SSP time discretization for solving hyperbolic problems. After all, they do not increase the computational cost and have the extra assurance of provable stability.

As we have already mentioned above, the high-order SSP methods discussed here are not restricted to preserving (not increasing) the TV. Our arguments below rely on convexity, hence these properties hold for *any* norm. Consequently, SSP methods have a wide range of applicability, as they can be used to ensure stability in an arbitrary norm once the forward Euler time discretization is shown to be strongly stable,¹ i.e., $\|u^n + \Delta t L(u^n)\| \leq \|u^n\|$. For linear examples we refer to [7], where weighted L^2 SSP higher order discretizations of spectral schemes were discussed. For nonlinear scalar conservation laws in several space dimensions, the TVD property is ruled out for high-resolution schemes; instead, strong stability in the maximum norm is sought. Applications of L^∞ SSP higher order discretization for discontinuous Galerkin and central schemes can be found in [3] and [9]. Finally, we note that since our arguments below are based on convex decompositions of high-order methods in terms of the first-order Euler method, any convex function will be preserved by such high-order time discretizations. In this context we refer, for example, to the cell entropy stability property of high-order schemes studied in [17] and [15].

We remark that the strong stability assumption for the forward Euler $\|u^n + \Delta t L(u^n)\| \leq \|u^n\|$ can be relaxed to the more general stability assumption $\|u^n + \Delta t L(u^n)\| \leq (1 + O(\Delta t))\|u^n\|$. This general stability property will also be preserved by the high-order SSP time discretizations. The total variation bounded (TVB) methods discussed in [18] and [2] belong to this category. However, if the forward Euler operator is not stable, the framework in this paper cannot be used to determine whether a high-order time discretization is stable or not.

2.2. SSP Runge–Kutta Methods. In [20], a general m -stage Runge–Kutta method for (2.1) is written in the form

$$(2.9) \quad \begin{aligned} u^{(0)} &= u^n, \\ u^{(i)} &= \sum_{k=0}^{i-1} \left(\alpha_{i,k} u^{(k)} + \Delta t \beta_{i,k} L(u^{(k)}) \right), \quad \alpha_{i,k} \geq 0, \quad i = 1, \dots, m, \\ u^{n+1} &= u^{(m)}. \end{aligned}$$

Clearly, if all the $\beta_{i,k}$'s are nonnegative, $\beta_{i,k} \geq 0$; then since by consistency $\sum_{k=0}^{i-1} \alpha_{i,k} = 1$, it follows that the intermediate stages in (2.9), $u^{(i)}$, amount to convex combinations of forward Euler operators, with Δt replaced by $\frac{\beta_{i,k}}{\alpha_{i,k}} \Delta t$. We thus conclude with the following lemma.

LEMMA 2.1 (see [20]). *If the forward Euler method (2.3) is strongly stable under the CFL restriction (2.4), $\|u^n + \Delta t L(u^n)\| \leq \|u^n\|$, then the Runge–Kutta method (2.9) with $\beta_{i,k} \geq 0$ is SSP, $\|u^{n+1}\| \leq \|u^n\|$, provided the following CFL restriction (2.6) is fulfilled:*

$$(2.10) \quad \Delta t \leq c \Delta t_{FE}, \quad c = \min_{i,k} \frac{\alpha_{i,k}}{\beta_{i,k}}.$$

¹By the notion of strong stability we refer to the fact that there is no temporal growth, as opposed to the general notion of stability, which allows a *bounded* temporal growth, $\|u^n\| \leq \text{Const} \cdot \|u^0\|$, with any arbitrary constant, possibly $\text{Const} > 1$.

If some of the $\beta_{i,k}$'s are negative, we need to introduce an associated operator \tilde{L} corresponding to stepping *backward* in time. The requirement for \tilde{L} is that it approximate the same spatial derivative(s) as L , but that the strong stability property hold with $\|u^{n+1}\| \leq \|u^n\|$ (with respect to either the TV or another relevant norm) for the first-order Euler scheme solved backward in time, i.e.,

$$(2.11) \quad u^{n+1} = u^n - \Delta t \tilde{L}(u^n).$$

This can be achieved for hyperbolic conservation laws by solving the negative-in-time version of (2.2),

$$(2.12) \quad u_t = f(u)_x.$$

Numerically, the only difference is the change of upwind direction. Clearly, \tilde{L} can be computed with the same cost as that of computing L . We then have the following lemma.

LEMMA 2.2 (see [20]). *If the forward Euler method combined with the spatial discretization L in (2.3) is strongly stable under the CFL restriction (2.4), $\|u^n + \Delta t L(u^n)\| \leq \|u^n\|$, and if Euler's method solved backward in time in combination with the spatial discretization \tilde{L} in (2.11) is also strongly stable under the CFL restriction (2.4), $\|u^n - \Delta t \tilde{L}(u^n)\| \leq \|u^n\|$, then the Runge–Kutta method (2.9) is SSP, $\|u^{n+1}\| \leq \|u^n\|$, under the CFL restriction (2.6),*

$$(2.13) \quad \Delta t \leq c \Delta t_{FE}, \quad c = \min_{i,k} \frac{\alpha_{i,k}}{|\beta_{i,k}|},$$

provided $\beta_{i,k} L$ is replaced by $\beta_{i,k} \tilde{L}$ whenever $\beta_{i,k}$ is negative.

Notice that if, for the same k , both $L(u^{(k)})$ and $\tilde{L}(u^{(k)})$ must be computed, the cost as well as the storage requirement for this k is doubled. For this reason, we would like to avoid negative $\beta_{i,k}$ as much as possible. However, as shown in [6] it is not always possible to avoid negative $\beta_{i,k}$.

2.3. SSP Multistep Methods.

SSP multistep methods of the form

$$(2.14) \quad u^{n+1} = \sum_{i=1}^m (\alpha_i u^{n+1-i} + \Delta t \beta_i L(u^{n+1-i})), \quad \alpha_i \geq 0,$$

were studied in [19]. Since $\sum \alpha_i = 1$, it follows that u^{n+1} is given by a convex combination of forward Euler solvers with suitably scaled Δt 's, and hence, similar to our discussion for Runge–Kutta methods, we arrive at the following lemma.

LEMMA 2.3 (see [19]). *If the forward Euler method combined with the spatial discretization L in (2.3) is strongly stable under the CFL restriction (2.4), $\|u^n + \Delta t L(u^n)\| \leq \|u^n\|$, and if Euler's method solved backward in time in combination with the spatial discretization \tilde{L} in (2.11) is also strongly stable under the CFL restriction (2.4), $\|u^n - \Delta t \tilde{L}(u^n)\| \leq \|u^n\|$, then the multistep method (2.14) is SSP, $\|u^{n+1}\| \leq \|u^n\|$, under the CFL restriction (2.6),*

$$(2.15) \quad \Delta t \leq c \Delta t_{FE}, \quad c = \min_i \frac{\alpha_i}{|\beta_i|},$$

provided $\beta_i L(\cdot)$ is replaced by $\beta_i \tilde{L}(\cdot)$ whenever β_i is negative.

3. Linear SSP Runge–Kutta Methods of Arbitrary Order.

3.1. SSP Runge–Kutta Methods with Optimal CFL Condition. In this section we present a class of optimal (in the sense of CFL number) SSP Runge–Kutta methods of any order for the ODE (2.1), where L is *linear*. With a linear L being realized as a finite-dimensional matrix, we denote $L(u) = Lu$. We will first show that the m -stage, m th-order SSP Runge–Kutta method can have, at most, CFL coefficient $c = 1$ in (2.10). We then proceed to construct optimal SSP linear Runge–Kutta methods.

PROPOSITION 3.1. *Consider the family of m -stage, m th-order SSP Runge–Kutta methods (2.9) with nonnegative coefficients $\alpha_{i,k}$ and $\beta_{i,k}$. The maximum CFL restriction attainable for such methods is the one dictated by the forward Euler scheme,*

$$\Delta t \leq \Delta t_{FE};$$

i.e., (2.6) holds with maximal CFL coefficient $c = 1$.

Proof. We consider the special case where L is linear and prove that even in this special case the maximum CFL coefficient c attainable is 1. Any m -stage method (2.9), for this linear case, can be rewritten as

$$u^{(i)} = \left(1 + \sum_{k=0}^{i-1} A_{i,k} (\Delta t L)^{k+1} \right) u^{(0)}, \quad i = 1, \dots, m,$$

where

$$\begin{aligned} A_{1,0} &= \beta_{1,0}, & A_{i,0} &= \sum_{k=1}^{i-1} \alpha_{i,k} A_{k,0} + \sum_{k=0}^{i-1} \beta_{i,k}, \\ A_{i,k} &= \sum_{j=k+1}^{i-1} \alpha_{i,j} A_{j,k} + \sum_{j=k}^{i-1} \beta_{i,j} A_{j,k-1}, & k &= 1, \dots, i-1. \end{aligned}$$

In particular, using induction, it is easy to show that the last two terms of the final stage can be expanded as

$$\begin{aligned} A_{m,m-1} &= \prod_{l=1}^m \beta_{l,l-1}, \\ A_{m,m-2} &= \sum_{k=2}^m \beta_{k,k-2} \left(\prod_{l=k+1}^m \beta_{l,l-1} \right) \left(\prod_{l=1}^{k-2} \beta_{l,l-1} \right) + \sum_{k=1}^m \alpha_{k,k-1} \left(\prod_{l=1, l \neq k}^m \beta_{l,l-1} \right). \end{aligned}$$

For an m -stage, m th-order linear Runge–Kutta scheme, $A_{m,k} = \frac{1}{(k+1)!}$. Using $A_{m,m-1} = \prod_{l=1}^m \beta_{l,l-1} = \frac{1}{m!}$, we can rewrite

$$A_{m,m-2} = \sum_{k=1}^m \frac{\alpha_{k,k-1}}{m! \beta_{k,k-1}} + \sum_{k=2}^m \beta_{k,k-2} \left(\prod_{l=k+1}^m \beta_{l,l-1} \right) \left(\prod_{l=1}^{k-2} \beta_{l,l-1} \right).$$

With the nonnegative assumption on $\beta_{i,k}$'s and the fact $A_{m,m-1} = \prod_{l=1}^m \beta_{l,l-1} = \frac{1}{m!}$ we have $\beta_{l,l-1} > 0$ for all l . For the CFL coefficient $c \geq 1$ we must have $\frac{\alpha_{k,k-1}}{\beta_{k,k-1}} \geq 1$ for all k . Clearly, $A_{m,m-2} = \frac{1}{(m-1)!}$ is possible under these restrictions only if $\beta_{k,k-2} = 0$ and $\frac{\alpha_{k,k-1}}{\beta_{k,k-1}} = 1$ for all k , in which case the CFL coefficient $c \leq 1$. \square

We remark that the conclusion of Proposition 3.1 is valid only if the m -stage Runge–Kutta method is m th-order accurate. In [19], we constructed an m -stage, first-order SSP Runge–Kutta method with a CFL coefficient $c = m$ which is suitable for steady state calculations.

The proof above also suggests a construction for the optimal linear m -stage, m th-order SSP Runge–Kutta methods.

PROPOSITION 3.2. *The class of m -stage schemes given (recursively) by*

$$(3.1) \quad u^{(i)} = u^{(i-1)} + \Delta t L u^{(i-1)}, \quad i = 1, \dots, m-1,$$

$$u^{(m)} = \sum_{k=0}^{m-2} \alpha_{m,k} u^{(k)} + \alpha_{m,m-1} \left(u^{(m-1)} + \Delta t L u^{(m-1)} \right),$$

where $\alpha_{1,0} = 1$ and

$$(3.2) \quad \alpha_{m,k} = \frac{1}{k} \alpha_{m-1,k-1}, \quad k = 1, \dots, m-2,$$

$$\alpha_{m,m-1} = \frac{1}{m!}, \quad \alpha_{m,0} = 1 - \sum_{k=1}^{m-1} \alpha_{m,k}$$

is an m th-order linear Runge–Kutta method which is SSP with CFL coefficient $c = 1$,

$$\Delta t \leq \Delta t_{FE}.$$

Proof. The first-order case is forward Euler, which is first-order accurate and SSP with CFL coefficient $c = 1$ by definition. The other schemes will be SSP with a CFL coefficient $c = 1$ by construction, as long as the coefficients are nonnegative.

We now show that scheme (3.1)–(3.2) is m th-order accurate when L is linear. In this case, clearly

$$u^{(i)} = (1 + \Delta t L)^i u^{(0)} = \left(\sum_{k=0}^i \frac{i!}{k!(i-k)!} (\Delta t L)^k \right) u^{(0)}, \quad i = 1, \dots, m-1,$$

hence scheme (3.1)–(3.2) results in

$$u^{(m)} = \left(\sum_{j=0}^{m-2} \alpha_{m,j} \sum_{k=0}^j \frac{j!}{k!(j-k)!} (\Delta t L)^k + \alpha_{m,m-1} \sum_{k=0}^m \frac{m!}{k!(m-k)!} (\Delta t L)^k \right) u^{(0)}.$$

Clearly, by (3.2), the coefficient of $(\Delta t L)^{m-1}$ is $\alpha_{m,m-1} \frac{m!}{(m-1)!} = \frac{1}{(m-1)!}$, the coefficient of $(\Delta t L)^m$ is $\alpha_{m,m-1} = \frac{1}{m!}$, and the coefficient of $(\Delta t L)^0$ is

$$\frac{1}{m!} + \sum_{j=0}^{m-2} \alpha_{m,j} = 1.$$

It remains to show that, for $1 \leq k \leq m-2$, the coefficient of $(\Delta t L)^k$ is equal to $\frac{1}{k!}$:

$$(3.3) \quad \frac{1}{k!(m-k)!} + \sum_{j=k}^{m-2} \alpha_{m,j} \frac{j!}{k!(j-k)!} = \frac{1}{k!}.$$

This will be shown by induction. Thus we assume (3.3) is true for m , and then for $m + 1$ we have, for $0 \leq k \leq m - 2$, that the coefficient of $(\Delta t L)^{k+1}$ is equal to

$$\begin{aligned}
& \frac{1}{(k+1)!(m-k)!} + \sum_{j=k+1}^{m-1} \alpha_{m+1,j} \frac{j!}{(k+1)!(j-k-1)!} \\
&= \frac{1}{(k+1)!} \left(\frac{1}{(m-k)!} + \sum_{l=k}^{m-2} \alpha_{m+1,l+1} \frac{(l+1)!}{(l-k)!} \right) \\
&= \frac{1}{(k+1)!} \left(\frac{1}{(m-k)!} + \sum_{l=k}^{m-2} \frac{1}{(l+1)} \alpha_{m,l} \frac{(l+1)!}{(l-k)!} \right) \\
&= \frac{1}{(k+1)!} \left(\frac{1}{(m-k)!} + \sum_{l=k}^{m-2} \alpha_{m,l} \frac{l!}{(l-k)!} \right) \\
&= \frac{1}{(k+1)!},
\end{aligned}$$

where in the second equality we used (3.2) and in the last equality we used the induction hypothesis (3.3). This finishes the proof.

Finally, we show that all the α 's are nonnegative. Clearly $\alpha_{2,0} = \alpha_{2,1} = \frac{1}{2} > 0$. If we assume $\alpha_{m,j} \geq 0$ for all $j = 0, \dots, m-1$, then

$$\alpha_{m+1,j} = \frac{1}{j} \alpha_{m,j-1} \geq 0, \quad j = 1, \dots, m-1; \quad \alpha_{m+1,m} = \frac{1}{(m+1)!} \geq 0,$$

and, by noticing that $\alpha_{m+1,j} \leq \alpha_{m,j-1}$ for all $j = 1, \dots, m$, we have

$$\alpha_{m+1,0} = 1 - \sum_{j=1}^m \alpha_{m+1,j} \geq 1 - \sum_{j=1}^m \alpha_{m,j-1} = 0. \quad \square$$

As the m -stage, m th-order linear Runge–Kutta method is unique, we have in effect proved that this unique m -stage, m th-order linear Runge–Kutta method is SSP under CFL coefficient $c = 1$. If L is nonlinear, scheme (3.1)–(3.2) is still SSP under CFL coefficient $c = 1$, but it is no longer m th-order accurate. Notice that all but the last stage of these methods are simple forward Euler steps.

We note in passing the examples of the ubiquitous third- and fourth-order Runge–Kutta methods, which admit the following convex, and hence SSP, decompositions:

$$(3.4) \quad \sum_{k=0}^3 \frac{1}{k!} (\Delta t L)^k = \frac{1}{3} + \frac{1}{2}(I + \Delta t L) + \frac{1}{6}(I + \Delta t L)^3,$$

$$(3.5) \quad \sum_{k=0}^4 \frac{1}{k!} (\Delta t L)^k = \frac{3}{8} + \frac{1}{3}(I + \Delta t L) + \frac{1}{4}(I + \Delta t L)^2 + \frac{1}{24}(I + \Delta t L)^4.$$

We list, in Table 3.1, the coefficients $\alpha_{m,j}$ of these optimal methods in (3.2) up to $m = 8$.

Table 3.1 Coefficients $\alpha_{m,j}$ of the SSP methods (3.1)–(3.2).

Order m	$\alpha_{m,0}$	$\alpha_{m,1}$	$\alpha_{m,2}$	$\alpha_{m,3}$	$\alpha_{m,4}$	$\alpha_{m,5}$	$\alpha_{m,6}$	$\alpha_{m,7}$
1	1							
2	$\frac{1}{2}$	$\frac{1}{2}$						
3	$\frac{1}{3}$	$\frac{1}{2}$	$\frac{1}{6}$					
4	$\frac{3}{8}$	$\frac{1}{3}$	$\frac{1}{4}$	$\frac{1}{24}$				
5	$\frac{11}{30}$	$\frac{3}{8}$	$\frac{1}{6}$	$\frac{1}{12}$	$\frac{1}{120}$			
6	$\frac{53}{144}$	$\frac{11}{30}$	$\frac{3}{16}$	$\frac{1}{18}$	$\frac{1}{48}$	$\frac{1}{720}$		
7	$\frac{103}{280}$	$\frac{53}{144}$	$\frac{11}{60}$	$\frac{3}{48}$	$\frac{1}{72}$	$\frac{1}{240}$	$\frac{1}{5040}$	
8	$\frac{2119}{5760}$	$\frac{103}{280}$	$\frac{53}{288}$	$\frac{11}{180}$	$\frac{1}{64}$	$\frac{1}{360}$	$\frac{1}{1440}$	$\frac{1}{40320}$

3.2. Application to Coercive Approximations. We now apply the optimal linear SSP Runge–Kutta methods to coercive approximations. We consider the linear system of ODEs of the general form, with possibly variable, time-dependent coefficients,

$$(3.6) \quad \frac{d}{dt}u(t) = L(t)u(t).$$

As an example we refer to [7], where the far-from-normal character of the spectral differentiation matrices defies the straightforward von Neumann stability analysis when augmented with high-order time discretizations.

We begin our stability study for Runge–Kutta approximations of (3.6) with the first-order forward Euler scheme (with $\langle \cdot, \cdot \rangle$ denoting the usual Euclidean inner product)

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

based on variable time steps, $t^n := \sum_{j=0}^{n-1} \Delta t_j$. Taking L^2 norms on both sides one finds

$$|u^{n+1}|^2 = |u^n|^2 + 2\Delta t_n \operatorname{Re}\langle L(t^n)u^n, u^n \rangle + (\Delta t_n)^2 |L(t^n)u^n|^2,$$

and hence strong stability holds, $|u^{n+1}| \leq |u^n|$, provided the following restriction on the time step, Δt_n , is met:

$$\Delta t_n \leq -2\operatorname{Re}\langle L(t^n)u^n, u^n \rangle / |L(t^n)u^n|^2.$$

Following Levy and Tadmor [13] we therefore make the following assumption.

ASSUMPTION 3.1 (coercivity). *The operator $L(t)$ is (uniformly) coercive in the sense that there exists $\eta(t) > 0$ such that*

$$(3.7) \quad \eta(t) := \inf_{|u|=1} -\frac{\operatorname{Re}\langle L(t)u, u \rangle}{|L(t)u|^2} > 0.$$

We conclude that for coercive L 's, the forward Euler scheme is strongly stable, $\|I + \Delta t_n L(t^n)\| \leq 1$, if and only if

$$\Delta t_n \leq 2\eta(t^n).$$

In a generic case, $L(t^n)$ represents a spatial operator with a coercivity bound $\eta(t^n)$, which is proportional to some power of the smallest spatial scale. In this context the above restriction on the time step amounts to the celebrated CFL stability condition. Our aim is to show that the general m -stage, m th-order accurate Runge–Kutta scheme is strongly stable under the *same* CFL condition.

Remark. Observe that the coercivity constant, η , is an upper bound in the size of L ; indeed, by Cauchy–Schwartz, $\eta(t) \leq |L(t)u| \cdot |u|/|L(t)u|^2$ and hence

$$(3.8) \quad \|L(t)\| = \sup_u \frac{|L(t)u|}{|u|} \leq \frac{1}{\eta(t)}.$$

To make one point, we consider the fourth-order Runge–Kutta approximation of (3.6):

$$(3.9) \quad k^1 = L(t^n)u^n,$$

$$(3.10) \quad k^2 = L(t^{n+\frac{1}{2}}) \left(u^n + \frac{\Delta t_n}{2} k^1 \right),$$

$$(3.11) \quad k^3 = L(t^{n+\frac{1}{2}}) \left(u^n + \frac{\Delta t_n}{2} k^2 \right),$$

$$(3.12) \quad k^4 = L(t^{n+1})(u^n + \Delta t_n k^3),$$

$$(3.13) \quad u^{n+1} = u^n + \frac{\Delta t_n}{6} [k^1 + 2k^2 + 2k^3 + k^4].$$

Starting with second-order and higher, the Runge–Kutta intermediate steps depend on the time variation of $L(\cdot)$, and hence we require a minimal smoothness in time, making the following assumption.

ASSUMPTION 3.2 (Lipschitz regularity). *We assume that $L(\cdot)$ is Lipschitz. Thus, there exists a constant $K > 0$ such that*

$$(3.14) \quad \|L(t) - L(s)\| \leq \frac{K}{\eta(t)} |t - s|.$$

We are now ready to make our main result, stating the following proposition.

PROPOSITION 3.3. *Consider the coercive systems of ODEs (3.6)–(3.7), with Lipschitz continuous coefficients (3.14). Then the fourth-order Runge–Kutta scheme (3.9)–(3.13) is stable under CFL condition*

$$(3.15) \quad \Delta t_n \leq 2\eta(t^n),$$

and the following estimate holds:

$$(3.16) \quad |u^n| \leq e^{3Kt_n} |u^0|.$$

Remark. The result along these lines was introduced by Levy and Tadmor [13, Main Theorem], stating the strong stability of the constant coefficients s -order Runge–Kutta scheme under CFL condition $\Delta t_n \leq C_s \eta(t^n)$. Here we improve in both simplicity and generality. Thus, for example, the previous bound of $C_4 = 1/31$ [13, Theorem 3.3] is now improved to a practical time-step restriction with our uniform $C_s = 2$.

Proof. We proceed in two steps. We first freeze the coefficients at $t = t^n$, considering (here we abbreviate $L^n = L(t^n)$)

$$(3.17) \quad j^1 = L^n u^n,$$

$$(3.18) \quad j^2 = L^n \left(u^n + \frac{\Delta t_n}{2} j^1 \right) \equiv L^n \left(I + \frac{\Delta t_n}{2} L^n \right) u^n,$$

$$(3.19) \quad j^3 = L^n \left(u^n + \frac{\Delta t_n}{2} j^2 \right) \equiv L^n \left[I + \frac{\Delta t_n}{2} L^n \left(I + \frac{\Delta t_n}{2} L^n \right) \right] u^n,$$

$$(3.20) \quad j^4 = L^n (u^n + \Delta t_n j^3),$$

$$(3.21) \quad v^{n+1} = u^n + \frac{\Delta t_n}{6} [j^1 + 2j^2 + 2j^3 + j^4].$$

Thus, $v^{n+1} = P_4(\Delta t_n L^n) u^n$, where following (3.5),

$$P_4(\Delta t_n L^n) := \frac{3}{8} I + \frac{1}{3} (I + \Delta t L) + \frac{1}{4} (I + \Delta t L)^2 + \frac{1}{24} (I + \Delta t L)^4.$$

Since the CFL condition (3.15) implies the strong stability of forward Euler, i.e., $\|I + \Delta t_n L^n\| \leq 1$, it follows that $\|P_4(\Delta t_n L^n)\| \leq 3/8 + 1/3 + 1/4 + 1/24 = 1$. Thus,

$$(3.22) \quad |v^{n+1}| \leq |u^n|.$$

Next, we include the time dependence. We need to measure the difference between the exact and the “frozen” intermediate values—the k ’s and the j ’s. We have

$$(3.23) \quad k^1 - j^1 = 0,$$

$$(3.24) \quad k^2 - j^2 = [L(t^{n+\frac{1}{2}}) - L(t^n)] \left(I + \frac{\Delta t_n}{2} L^n \right) u^n,$$

$$(3.25) \quad k^3 - j^3 = L(t^{n+\frac{1}{2}}) \frac{\Delta t_n}{2} (k^2 - j^2) + [L(t^{n+\frac{1}{2}}) - L(t^n)] \frac{\Delta t_n}{2} j^2,$$

$$(3.26) \quad k^4 - j^4 = L(t^{n+1}) \Delta t_n (k^3 - j^3) + [L(t^{n+1}) - L(t^n)] \Delta t_n j^3.$$

Lipschitz continuity (3.14) and the strong stability of forward Euler imply

$$(3.27) \quad |k^2 - j^2| \leq \frac{K \cdot \Delta t_n}{2\eta(t^n)} |u^n| \leq K |u^n|.$$

Also, since $\|L^n\| \leq \frac{1}{\eta(t^n)}$, we find from (3.18) that $|j^2| \leq |u^n|/\eta(t^n)$, and hence (3.25) followed by (3.27) and the CFL condition (3.15) imply

$$(3.28) \quad \begin{aligned} |k^3 - j^3| &\leq \frac{\Delta t_n}{2\eta(t^n)} |k^2 - j^2| + \frac{K \cdot \Delta t_n}{2\eta(t^n)} \cdot \frac{\Delta t_n}{2\eta(t^n)} |u^n| \\ &\leq 2K \left(\frac{\Delta t_n}{2\eta(t^n)} \right)^2 |u^n| \leq 2K |u^n|. \end{aligned}$$

Finally, since by (3.19) j^3 does not exceed $|j^3| < \frac{1}{\eta(t^n)}(1 + \frac{\Delta t_n}{2\eta(t^n)})|u^n|$, we find from (3.26) followed by (3.28) and the CFL condition (3.15),

$$(3.29) \quad |k^4 - j^4| \leq \frac{\Delta t_n}{\eta(t^n)} |k^3 - j^3| + \frac{K \cdot \Delta t_n}{\eta(t^n)} \cdot \frac{\Delta t_n}{\eta(t^n)} \left(1 + \frac{\Delta t_n}{2\eta(t^n)}\right) |u^n|$$

$$\leq K \left(\left(\frac{\Delta t_n}{\eta(t^n)}\right)^3 + \left(\frac{\Delta t_n}{\eta(t^n)}\right)^2 \right) |u^n| \leq 12K |u^n|.$$

We conclude that u^{n+1} ,

$$u^{n+1} = v^{n+1} + \frac{\Delta t_n}{6} \left[2(k^2 - j^2) + 2(k^3 - j^3) + (k^4 - j^4) \right],$$

is upper bounded by (consult (3.22), (3.27)–(3.29))

$$|u^{n+1}| \leq |v^{n+1}| + \frac{\Delta t_n}{6} \left[2K|u^n| + 4K|u^n| + 12K|u^n| \right]$$

$$\leq (1 + 3K\Delta t_n) |u^n|$$

and the result (3.16) follows. \square

4. Nonlinear SSP Runge–Kutta Methods. In the previous section we derived SSP Runge–Kutta methods for linear spatial discretizations. As explained in the introduction, SSP methods are often required for nonlinear spatial discretizations. Thus, most of the research to date has been in the derivation of SSP methods for nonlinear spatial discretizations. In [20], schemes up to third order were found to satisfy the conditions in Lemma 2.1 with CFL coefficient $c = 1$. In [6] it was shown that all four-stage, fourth-order Runge–Kutta methods with positive CFL coefficient c in (2.13) must have at least one negative $\beta_{i,k}$, and a method which seems optimal was found. For large-scale scientific computing in three space dimensions, storage is usually a paramount consideration. We review the results presented in [6] about SSP properties among such low-storage Runge–Kutta methods.

4.1. Nonlinear Methods of Second, Third, and Fourth Order. Here we review the optimal (in the sense of CFL coefficient and the cost incurred by \tilde{L} if it appears) SSP Runge–Kutta methods of m -stage, m th-order for $m = 2, 3, 4$, written in the form (2.9).

PROPOSITION 4.1 (see [6]). *If we require $\beta_{i,k} \geq 0$, then an optimal second-order SSP Runge–Kutta method (2.9) is given by*

$$(4.1) \quad u^{(1)} = u^n + \Delta t L(u^n),$$

$$u^{n+1} = \frac{1}{2}u^n + \frac{1}{2}u^{(1)} + \frac{1}{2}\Delta t L(u^{(1)}),$$

with a CFL coefficient $c = 1$ in (2.10). An optimal third-order SSP Runge–Kutta method (2.9) is given by

$$(4.2) \quad u^{(1)} = u^n + \Delta t L(u^n),$$

$$u^{(2)} = \frac{3}{4}u^n + \frac{1}{4}u^{(1)} + \frac{1}{4}\Delta t L(u^{(1)}),$$

$$u^{n+1} = \frac{1}{3}u^n + \frac{2}{3}u^{(2)} + \frac{2}{3}\Delta t L(u^{(2)}),$$

with a CFL coefficient $c = 1$ in (2.10).

In the fourth-order case we proved in [6] that we cannot avoid the appearance of negative $\beta_{i,k}$, as demonstrated in the following proposition.

PROPOSITION 4.2 (see [6]). *The four-stage, fourth-order SSP Runge–Kutta scheme (2.9) with a nonzero CFL coefficient c in (2.13) must have at least one negative $\beta_{i,k}$.*

We thus must settle for finding an efficient fourth-order scheme containing \tilde{L} , which maximizes the operation cost measured by $\frac{c}{4+i}$, where c is the CFL coefficient (2.13) and i is the number of \tilde{L} 's. This way we are looking for an SSP method that reaches a fixed time T with a minimal number of evaluations of L or \tilde{L} . The best method we could find in [6] is

$$\begin{aligned}
 u^{(1)} &= u^n + \frac{1}{2}\Delta t L(u^n), \\
 u^{(2)} &= \frac{649}{1600}u^{(0)} - \frac{10890423}{25193600}\Delta t \tilde{L}(u^n) + \frac{951}{1600}u^{(1)} + \frac{5000}{7873}\Delta t L(u^{(1)}), \\
 (4.3) \quad u^{(3)} &= \frac{53989}{2500000}u^n - \frac{102261}{5000000}\Delta t \tilde{L}(u^n) + \frac{4806213}{20000000}u^{(1)} \\
 &\quad - \frac{5121}{20000}\Delta t \tilde{L}(u^{(1)}) + \frac{23619}{32000}u^{(2)} + \frac{7873}{10000}\Delta t L(u^{(2)}), \\
 u^{n+1} &= \frac{1}{5}u^n + \frac{1}{10}\Delta t L(u^n) + \frac{6127}{30000}u^{(1)} + \frac{1}{6}\Delta t L(u^{(1)}) \\
 &\quad + \frac{7873}{30000}u^{(2)} + \frac{1}{3}u^{(3)} + \frac{1}{6}\Delta t L(u^{(3)}),
 \end{aligned}$$

with a CFL coefficient $c = 0.936$ in (2.13). Notice that two \tilde{L} 's must be computed. The effective CFL coefficient, compared with an ideal case without \tilde{L} 's, is $0.936 \times \frac{4}{6} = 0.624$. Since it is difficult to solve the global optimization problem, we do not claim that (4.3) is an optimal four stage, fourth-order SSP Runge–Kutta method.

4.2. Low Storage Methods. Storage is usually an important consideration for large scale scientific computing in three space dimensions. Therefore, low-storage Runge–Kutta methods [23], [1], which only require two storage units per ODE variable, may be desirable. Here we review the results presented in [6] concerning SSP properties among such low-storage Runge–Kutta methods.

The general low-storage Runge–Kutta schemes can be written in the form [23], [1]

$$\begin{aligned}
 u^{(0)} &= u^n, \quad du^{(0)} = 0, \\
 du^{(i)} &= A_i du^{(i-1)} + \Delta t L(u^{(i-1)}), \quad i = 1, \dots, m, \\
 (4.4) \quad u^{(i)} &= u^{(i-1)} + B_i du^{(i)}, \quad i = 1, \dots, m, \quad B_1 = c, \\
 u^{n+1} &= u^{(m)}.
 \end{aligned}$$

Only u and du must be stored, resulting in two storage units for each variable.

Following Carpenter and Kennedy [1], the best SSP third-order method found by numerical search in [6] is given by the system

$$\begin{aligned}
 z_1 &= \sqrt{36c^4 + 36c^3 - 135c^2 + 84c - 12}, & z_2 &= 2c^2 + c - 2, \\
 z_3 &= 12c^4 - 18c^3 + 18c^2 - 11c + 2, & z_4 &= 36c^4 - 36c^3 + 13c^2 - 8c + 4, \\
 z_5 &= 69c^3 - 62c^2 + 28c - 8, & z_6 &= 34c^4 - 46c^3 + 34c^2 - 13c + 2,
 \end{aligned}$$

$$\begin{aligned}
B_2 &= \frac{12c(c-1)(3z_2 - z_1) - (3z_2 - z_1)^2}{144c(3c-2)(c-1)^2}, \\
B_3 &= \frac{-24(3c-2)(c-1)^2}{(3z_2 - z_1)^2 - 12c(c-1)(3z_2 - z_1)}, \\
A_2 &= \frac{-z_1(6c-4c+1) + 3z_3}{(2c+1)z_1 - 3(c+2)(2c-1)^2}, \\
A_3 &= \frac{-z_1z_4 + 108(2c-1)c^5 - 3(2c-1)z_5}{24z_1c(c-1)^4 + 72cz_6 + 72c^6(2c-13)},
\end{aligned}$$

with $c = 0.924574$, resulting in a CFL coefficient $c = 0.32$ in (2.6). This is of course less optimal than (4.2) in terms of CFL coefficient, but the low-storage form is useful for large-scale calculations. Carpenter and Kennedy [1] have also given classes of five-stage, fourth-order low-storage Runge–Kutta methods. We have been unable to find SSP methods in that class with positive $\alpha_{i,k}$ and $\beta_{i,k}$. A low-storage method with negative $\beta_{i,k}$ cannot be made SSP, as \tilde{L} cannot be used without destroying the low-storage property.

4.3. Hybrid Multistep Runge–Kutta Methods. Hybrid multistep Runge–Kutta methods (e.g., [10], [14]) are methods that combine the properties of Runge–Kutta and multistep methods. We explore the two-step, two-stage method

$$\begin{aligned}
(4.5) \quad u^{n+\frac{1}{2}} &= \alpha_{21}u^n + \alpha_{20}u^{n-1} + \Delta t (\beta_{20}L(u^{n-1}) + \beta_{21}L(u^n)), \quad \alpha_{2k} \geq 0, \\
u^{n+1} &= \alpha_{30}u^{n-1} + \alpha_{31}u^{n+\frac{1}{2}} + \alpha_{32}u^n \\
(4.6) \quad &+ \Delta t (\beta_{30}L(u^{n-1}) + \beta_{31}L(u^{n+\frac{1}{2}}) + \beta_{32}L(u^n)), \quad \alpha_{3k} \geq 0.
\end{aligned}$$

Clearly, this method is SSP under the CFL coefficient (2.10) if $\beta_{i,k} \geq 0$. We could also consider the case allowing negative $\beta_{i,k}$'s, using instead (2.13) for the CFL coefficient and replacing $\beta_{i,k}L$ by $\beta_{i,k}\tilde{L}$ for the negative $\beta_{i,k}$'s.

For third-order accuracy, we have a three-parameter family (depending on c , α_{30} , and α_{31}):

$$\begin{aligned}
(4.7) \quad \alpha_{20} &= 3c^2 + 2c^3, \\
\beta_{20} &= c^2 + c^3, \\
\alpha_{21} &= 1 - 3c^2 - 2c^3, \\
\beta_{21} &= c + 2c^2 + c^3, \\
\beta_{30} &= \frac{2 + 2\alpha_{30} - 3c + 3\alpha_{30}c + \alpha_{31}c^3}{6(1+c)}, \\
\beta_{31} &= \frac{5 - \alpha_{30} - 3\alpha_{31}c^2 - 2\alpha_{31}c^3}{6c + 6c^2}, \\
\alpha_{32} &= 1 - \alpha_{31} - \alpha_{30}, \\
\beta_{32} &= \frac{-5 + \alpha_{30} + 9c + 3\alpha_{30}c - 3\alpha_{31}c^2 - \alpha_{31}c^3}{6c}.
\end{aligned}$$

The best method we were able to find is given by $c = 0.4043$, $\alpha_{30} = 0.0605$, and $\alpha_{31} = 0.6315$ and has a CFL coefficient $c \approx 0.473$. Clearly, this is not as good as the optimal third-order Runge–Kutta method (4.2) with CFL coefficient $c = 1$. We hoped that a fourth-order scheme with a large CFL coefficient could be found, but unfortunately this is not the case, as is proven in the following proposition.

PROPOSITION 4.3. *There are no fourth-order schemes (4.5) with all nonnegative $\alpha_{i,k}$.*

Proof. The fourth-order schemes are given by a two-parameter family depending on c, α_{30} and setting α_{31} in (4.7) to be

$$\alpha_{31} = \frac{-7 - \alpha_{30} + 10c - 2\alpha_{30}c}{c^2(3 + 8c + 4c^2)}.$$

The requirement $\alpha_{21} \geq 0$ enforces (see (4.7)) $c \leq \frac{1}{2}$. The further requirement $\alpha_{20} \geq 0$ yields $-\frac{3}{2} \leq c \leq \frac{1}{2}$. α_{31} has a positive denominator and a negative numerator for $-\frac{1}{2} < c < \frac{1}{2}$, and its denominator is 0 when $c = -\frac{1}{2}$ or $c = -\frac{3}{2}$, thus we require $-\frac{3}{2} \leq c < -\frac{1}{2}$. In this range, the denominator of α_{31} is negative, hence we also require its numerator to be negative, which translates to $\alpha_{30} \leq \frac{-7+10c}{1+2c}$. Finally, we would require $\alpha_{32} = 1 - \alpha_{31} - \alpha_{30} \geq 0$, which translates to $\alpha_{30} \geq \frac{c^2(2c+1)(2c+3)+7-10c}{(2c+1)(2c-1)(c+1)^2}$. The two restrictions on α_{30} give us the following inequality:

$$\frac{-7 + 10c}{1 + 2c} \geq \frac{c^2(2c + 1)(2c + 3) + 7 - 10c}{(2c + 1)(2c - 1)(c + 1)^2},$$

which, in the range of $-\frac{3}{2} \leq c < -\frac{1}{2}$, yields $c \geq 1$ —a contradiction. \square

5. Linear and Nonlinear Multistep Methods. In this section we review and further study SSP explicit multistep methods (2.14), which were first developed in [19]. These methods are r th-order accurate if

$$(5.1) \quad \sum_{i=1}^m \alpha_i = 1,$$

$$\sum_{i=1}^m i^k \alpha_i = k \left(\sum_{i=1}^m i^{k-1} \beta_i \right), \quad k = 1, \dots, r.$$

We first prove a proposition that sets the minimum number of steps in our search for SSP multistep methods.

PROPOSITION 5.1. *For $m \geq 2$, there is no m -step, m th-order SSP method with all nonnegative β_i , and there is no m -step SSP method of order $(m + 1)$.*

Proof. By the accuracy condition (5.1), we clearly have for an r th-order accurate method

$$(5.2) \quad \sum_{i=1}^m p(i)\alpha_i = \sum_{i=1}^m p'(i)\beta_i$$

for any polynomial $p(x)$ of degree at most r satisfying $p(0) = 0$.

When $r = m$, we could choose

$$p(x) = x(m - x)^{m-1}.$$

Clearly $p(i) \geq 0$ for $i = 1, \dots, m$, and equality holds only for $i = m$. On the other hand, $p'(i) = m(1-i)(m-i)^{m-2} \leq 0$, and equality holds only for $i = 1$ and $i = m$. Hence (5.2) would have a negative right side and a positive left side and would not be an inequality if all α_i and β_i are nonnegative, unless the only nonzero entries are α_m , β_1 , and β_m . In this special case we have $\alpha_m = 1$ and $\beta_1 = 0$ to get a positive CFL coefficient c in (2.15). The first two-order conditions in (5.1) now lead to $\beta_m = m$ and $2\beta_m = m$, which cannot be simultaneously satisfied.

When $r = m + 1$, we could choose

$$(5.3) \quad p(x) = \int_0^x q(t) dt, \quad q(t) = \prod_{i=1}^m (i-t).$$

Clearly $p'(i) = q(i) = 0$ for $i = 1, \dots, m$. We also claim (and prove below) that all the $p(i)$'s, $i = 1, \dots, m$, are positive. With this choice of p in (5.2), its right-hand side vanishes, while the left-hand side is strictly positive if all $\alpha_i \geq 0$ —a contradiction.

We conclude with the proof of the following claim.

Claim. $p(i) = \int_0^i q(t) dt > 0$, $q(t) := \prod_{i=1}^m (i-t)$.

Indeed, $q(t)$ oscillates between being positive on the even intervals $I_0 = (0, 1)$, $I_2 = (2, 3), \dots$, and being negative on the odd intervals, $I_1 = (1, 2)$, $I_3 = (3, 4), \dots$. The positivity of the $p(i)$'s for $i \leq (m+1)/2$ follows since the integral of $q(t)$ over each pair of consecutive intervals is positive, at least for the first $\lfloor (m+1)/2 \rfloor$ intervals,

$$\begin{aligned} p(2k+2) - p(2k) &= \int_{I_{2k}} |q(t)| dt - \int_{I_{2k+1}} |q(t)| dt \\ &= \int_{I_{2k}} - \int_{I_{2k+1}} |(1-t)(2-t) \cdots (m-t)| dt \\ &= \int_{I_{2k}} |(1-t)(2-t) \cdots (m-1-t)| \times (|(m-t)| - |t|) dt > 0, \\ & \qquad \qquad \qquad 2k+1 \leq (m+1)/2. \end{aligned}$$

For the remaining intervals we note the symmetry of $q(t)$ with respect to the midpoint $(m+1)/2$, i.e., $q(t) = (-1)^m q(m+1-t)$, which enables us to write for $i > (m+1)/2$

$$(5.4) \quad \begin{aligned} p(i) &= \int_0^{(m+1)/2} q(t) dt + (-1)^m \int_{(m+1)/2}^i q(m+1-t) dt \\ &= \int_0^{(m+1)/2} q(t) dt + (-1)^m \int_{m+1-i}^{(m+1)/2} q(t') dt'. \end{aligned}$$

Thus, if m is odd, then $p(i) = p(m+1-i) > 0$ for $i > (m+1)/2$. If m is even, then the second integral on the right of (5.4) is positive for odd i 's, since it starts with a positive integrand on the even interval I_{m+1-i} . And finally, if m is even and i is odd, then the second integral starts with a negative contribution from its first integrand on the odd interval I_{m+1-i} , while the remaining terms that follow cancel in pairs as before. A straightforward computation shows that this first negative contribution is compensated for by the positive gain from the first pair, i.e.,

$$p(m+2-i) > \int_0^2 q(t) dt + \int_{m+1-i}^{m+2-i} q(t) dt > 0, \quad m \text{ even, } i \text{ odd.}$$

This concludes the proof of our claim. \square

Table 5.1 *SSP multistep methods (2.14).*

#	Steps m	Order r	CFL c	α_i	β_i
1	2	2	$\frac{1}{2}$	$\frac{4}{5}, \frac{1}{5}$	$\frac{8}{5}, -\frac{2}{5}$
2	3	2	$\frac{1}{2}$	$\frac{3}{4}, 0, \frac{1}{4}$	$\frac{3}{2}, 0, 0$
3	4	2	$\frac{2}{3}$	$\frac{8}{9}, 0, 0, \frac{1}{9}$	$\frac{4}{3}, 0, 0, 0$
4	3	3	0.274	$\frac{4}{7}, \frac{2}{7}, \frac{1}{7}$	$\frac{25}{12}, -\frac{20}{21}, \frac{37}{84}$
5	3	3	0.287	$\frac{2973}{5000}, \frac{351}{1250}, \frac{623}{5000}$	$\frac{1297}{625}, -\frac{49}{50}, \frac{1087}{2500}$
6	4	3	$\frac{1}{3}$	$\frac{16}{27}, 0, 0, \frac{11}{27}$	$\frac{16}{9}, 0, 0, \frac{4}{9}$
7	5	3	$\frac{1}{2}$	$\frac{25}{32}, 0, 0, 0, \frac{7}{32}$	$\frac{25}{16}, 0, 0, 0, \frac{5}{16}$
8	6	3	0.567	$\frac{108}{125}, 0, 0, 0, 0, \frac{17}{125}$	$\frac{36}{25}, 0, 0, 0, 0, \frac{6}{25}$
9	4	4	0.154	$\frac{29}{72}, \frac{7}{24}, \frac{1}{4}, \frac{1}{18}$	$\frac{481}{192}, -\frac{1055}{576}, \frac{937}{576}, -\frac{197}{576}$
10	4	4	0.159	$\frac{1989}{5000}, \frac{2893}{10000}, \frac{517}{2000}, \frac{34}{625}$	$\frac{601613}{240000}, -\frac{1167}{640}, \frac{130301}{80000}, -\frac{82211}{240000}$
11	6	4	0.245	$\frac{747}{1280}, 0, 0, 0, \frac{81}{256}, \frac{1}{10}$	$\frac{237}{128}, 0, 0, 0, \frac{165}{128}, -\frac{3}{8}$
12	5	4	0.021	$\frac{1557}{32000}, \frac{1}{32000}, \frac{1}{120}, \frac{2063}{48000}, \frac{9}{10}$	$\frac{5323561}{2304000}, \frac{2659}{2304000}, \frac{904987}{2304000}, \frac{1567579}{768000}, 0$
13	5	5	0.077	$\frac{1}{4}, \frac{1}{4}, \frac{7}{24}, \frac{1}{6}, \frac{1}{24}$	$\frac{185}{64}, -\frac{851}{288}, \frac{91}{24}, -\frac{151}{96}, \frac{199}{576}$
14	5	5	0.085	$\frac{1}{4}, \frac{13}{50}, \frac{8}{25}, \frac{7}{50}, \frac{3}{100}$	$\frac{52031}{18000}, -\frac{26617}{9000}, \frac{1412}{375}, -\frac{14407}{9000}, \frac{6161}{18000}$
15	6	5	0.130	$\frac{7}{20}, \frac{3}{10}, \frac{4}{15}, 0, \frac{7}{120}, \frac{1}{40}$	$\frac{291201}{108000}, -\frac{198401}{86400}, \frac{88063}{43200}, 0, -\frac{17969}{43200}, \frac{73061}{432000}$

We remark that [4] contains a result stating that there are no linearly stable m -step, $(m + 1)$ st-order methods when m is odd. When m is even such linearly stable methods exist but would require negative α_i . This is consistent with our result.

In the remainder of this section we will discuss optimal m -step, m th-order SSP methods (which must have negative β_i according to Proposition 5.1) and m -step, $(m - 1)$ st-order SSP methods with positive β_i .

For two-step, second-order SSP methods, a scheme was given in [19] with a CFL coefficient $c = \frac{1}{2}$ (scheme 1 in Table 5.1). We prove this is optimal in terms of CFL coefficients.

PROPOSITION 5.2. *For two-step, second-order SSP methods, the optimal CFL coefficient c in (2.15) is $\frac{1}{2}$.*

Proof. The accuracy condition (5.1) can be explicitly solved to obtain a one-parameter family of solutions

$$\alpha_2 = 1 - \alpha_1, \quad \beta_1 = 2 - \frac{1}{2}\alpha_1, \quad \beta_2 = -\frac{1}{2}\alpha_1.$$

The CFL coefficient c is a function of α_1 and it can be easily verified that the maximum is $c = \frac{1}{2}$ achieved at $\alpha_1 = \frac{4}{5}$. \square

We move on to three-step, second-order methods. It is now possible to have SSP schemes with positive α_i and β_i . One such method is given in [19] with a CFL coefficient $c = \frac{1}{2}$ (scheme 2 in Table 5.1). We prove this is optimal in the CFL coefficient in the following proposition. We remark that this multistep method has the same efficiency as the optimal two-stage, second-order Runge–Kutta method (4.1). This is because there is only one L evaluation per time step here, compared with two L evaluations in the two-stage Runge–Kutta method. Of course, the storage requirement here is larger.

PROPOSITION 5.3. *If we require $\beta_i \geq 0$, then the optimal three-step, second-order method has a CFL coefficient $c = \frac{1}{2}$.*

Proof. The coefficients of the three-step, second-order method are given by

$$\alpha_1 = \frac{1}{2}(6 - 3\beta_1 - \beta_2 + \beta_3), \quad \alpha_2 = -3 + 2\beta_1 - 2\beta_3, \quad \alpha_3 = \frac{1}{2}(2 - \beta_1 + \beta_2 + 3\beta_3).$$

For CFL coefficient $c > \frac{1}{2}$, we need $\frac{\alpha_k}{\beta_k} > \frac{1}{2}$ for all k . This implies

$$\begin{aligned} 2\alpha_1 > \beta_1 &\Rightarrow 6 - 4\beta_1 - \beta_2 + \beta_3 > 0, \\ 2\alpha_2 > \beta_2 &\Rightarrow -6 + 4\beta_1 - \beta_2 - 4\beta_3 > 0. \end{aligned}$$

This means that

$$\beta_2 - \beta_3 < 6 - 4\beta_1 < -\beta_2 - 4\beta_3 \quad \Rightarrow \quad 2\beta_2 < -3\beta_3.$$

Thus, we would have a negative β . \square

We remark that if more steps are allowed, then the CFL coefficient can be improved. Scheme 3 in Table 5.1 is a four-step, second-order method with positive α_i and β_i and a CFL coefficient $c = \frac{2}{3}$.

We now move to three-step, third-order methods. In [19] we gave a three-step, third-order method with a CFL coefficient $c \approx 0.274$ (scheme 4 in Table 5.1). A computer search gives a slightly better scheme (scheme 5 in Table 5.1) with a CFL coefficient $c \approx 0.287$.

Next we move on to four-step, third-order methods. It is now possible to have SSP schemes with positive α_i and β_i . One example was given in [19] with a CFL coefficient $c = \frac{1}{3}$ (scheme 6 in Table 5.1). We prove this is optimal in the CFL coefficient in the following proposition. We remark again that this multistep method has the same efficiency as the optimal three-stage, third-order Runge–Kutta method (4.2). This is because there is only one L evaluation per time step here, compared with three L evaluations in the three-stage Runge–Kutta method. Of course, the storage requirement here is larger.

PROPOSITION 5.4. *If we require $\beta_i \geq 0$, then the optimal four-step, third-order method has a CFL coefficient $c = \frac{1}{3}$.*

Proof. The coefficients of the four-step, third-order method are given by

$$\begin{aligned} \alpha_1 &= \frac{1}{6}(24 - 11\beta_1 - 2\beta_2 + \beta_3 - 2\beta_4), & \alpha_2 &= -6 + 3\beta_1 - \frac{1}{2}\beta_2 - \beta_3 + \frac{3}{2}\beta_4, \\ \alpha_3 &= 4 - \frac{3}{2}\beta_1 + \beta_2 + \frac{1}{2}\beta_3 - 3\beta_4, & \alpha_4 &= \frac{1}{6}(-6 + 2\beta_1 - \beta_2 + 2\beta_3 + 11\beta_4). \end{aligned}$$

For a CFL coefficient $c > \frac{1}{3}$ we need $\frac{\alpha_k}{\beta_k} > \frac{1}{3}$ for all k . This implies

$$\begin{aligned} 24 - 13\beta_1 - 2\beta_2 + \beta_3 - 2\beta_4 &> 0, & -36 + 18\beta_1 - 5\beta_2 - 6\beta_3 + 9\beta_4 &> 0, \\ 24 - 9\beta_1 + 6\beta_2 + \beta_3 - 18\beta_4 &> 0, & -6 + 2\beta_1 - \beta_2 + 2\beta_3 + 9\beta_4 &> 0. \end{aligned}$$

Combining these (9 times the first inequality plus 8 times the second plus 3 times the third) we get

$$-40\beta_2 - 36\beta_3 > 0,$$

which implies a negative β . \square

We again remark that if more steps are allowed, the CFL coefficient can be improved. Scheme 7 in Table 5.1 is a five-step, third-order method with positive α_i and β_i and a CFL coefficient $c = \frac{1}{2}$. Scheme 8 in Table 5.1 is a six-step, third-order method with positive α_i and β_i and a CFL coefficient $c = 0.567$.

We now move on to four-step, fourth-order methods. In [19] we gave a four-step, fourth-order method (scheme 9 in Table 5.1) with a CFL coefficient $c \approx 0.154$. A computer search gives a slightly better scheme with a CFL coefficient $c \approx 0.159$, scheme 10 in Table 5.1. If we allow two more steps, we can improve the CFL coefficient to $c = 0.245$ (scheme 11 in Table 5.1).

Next we move on to five-step, fourth-order methods. It is now possible to have SSP schemes with positive α_i and β_i . The solution can be written as the following five-parameter family:

$$\begin{aligned} \alpha_5 &= 1 - \alpha_1 - \alpha_2 - \alpha_3 - \alpha_4, & \beta_1 &= \frac{1}{24} (55 + 9\alpha_2 + 8\alpha_3 + 9\alpha_4 + 24\beta_5), \\ \beta_2 &= \frac{1}{24} (5 - 64\alpha_1 - 45\alpha_2 - 32\alpha_3 - 37\alpha_4 - 96\beta_5), \\ \beta_3 &= \frac{1}{24} (5 + 32\alpha_1 + 27\alpha_2 + 40\alpha_3 + 59\alpha_4 + 144\beta_5), \\ \beta_4 &= \frac{1}{24} (55 - 64\alpha_1 - 63\alpha_2 - 64\alpha_3 - 55\alpha_4 - 96\beta_5). \end{aligned}$$

We can clearly see that to get $\beta_2 \geq 0$ we would need $\alpha_1 \leq \frac{5}{64}$, and also $\beta_1 \geq \frac{55}{24}$, hence the CFL coefficient cannot exceed $c \leq \frac{\alpha_1}{\beta_1} \leq \frac{3}{88} \approx 0.034$. A computer search gives a scheme (scheme 12 in Table 5.1) with a CFL coefficient $c = 0.021$. The significance of this scheme is that it disproves the belief that SSP schemes of order four or higher must have negative β and hence must use \tilde{L} (see Proposition 4.2 for Runge–Kutta methods). However, the CFL coefficient here is probably too small for the scheme to be of much practical use.

We finally look at five-step, fifth-order methods. In [19] a scheme with CFL coefficient $c = 0.077$ is given (scheme 13 in Table 5.1). A computer search gives us a scheme with a slightly better CFL coefficient $c \approx 0.085$, scheme 14 in Table 5.1. Finally, by increasing one more step, one could get [19], a scheme with CFL coefficient $c = 0.130$, scheme 15 in Table 5.1.

We list in Table 5.1 the multistep methods studied in this section.

6. Implicit SSP Methods.

6.1. Implicit TVD Stable Scheme. Implicit methods are useful in that they typically eliminate the step-size restriction (CFL) associated with stability analysis. For many applications, the backward Euler method possesses strong stability properties that we would like to preserve in higher order methods. For example, it is easy to show a version of Harten’s lemma [8] for the TVD property of implicit backward Euler methods.

LEMMA 6.1 (Harten). *The following implicit backward Euler method*

$$(6.1) \quad u_j^{n+1} = u_j^n + \Delta t \left[C_{j+\frac{1}{2}} (u_{j+1}^{n+1} - u_j^{n+1}) - D_{j-\frac{1}{2}} (u_j^{n+1} - u_{j-1}^{n+1}) \right],$$

where $C_{j+\frac{1}{2}}$ and $D_{j-\frac{1}{2}}$ are functions of u^n and/or u^{n+1} at various (usually neighboring) grid points satisfying

$$(6.2) \quad C_{j+\frac{1}{2}} \geq 0, \quad D_{j-\frac{1}{2}} \geq 0,$$

is TVD in the sense of (2.5) for arbitrary Δt .

Proof. Taking a spatial forward difference in (6.1) and moving terms, one gets

$$\begin{aligned} & \left[1 + \Delta t \left(C_{j+\frac{1}{2}} + D_{j+\frac{1}{2}}\right)\right] (u_{j+1}^{n+1} - u_j^{n+1}) \\ &= u_{j+1}^n - u_j^n + \Delta t C_{j+\frac{3}{2}} (u_{j+2}^{n+1} - u_{j+1}^{n+1}) + \Delta t D_{j-\frac{1}{2}} (u_j^{n+1} - u_{j-1}^{n+1}). \end{aligned}$$

Using the positivity of C and D in (6.2) one gets

$$\begin{aligned} & \left[1 + \Delta t \left(C_{j+\frac{1}{2}} + D_{j+\frac{1}{2}}\right)\right] |u_{j+1}^{n+1} - u_j^{n+1}| \\ & \leq |u_{j+1}^n - u_j^n| + \Delta t C_{j+\frac{3}{2}} |u_{j+2}^{n+1} - u_{j+1}^{n+1}| + \Delta t D_{j-\frac{1}{2}} |u_j^{n+1} - u_{j-1}^{n+1}|, \end{aligned}$$

which, upon summing over j , would yield the TVD property (2.5). \square

Another example is the cell entropy inequality for the square entropy, satisfied by the discontinuous Galerkin method of arbitrary order of accuracy in any space dimensions, when the time discretization is by a class of implicit time discretization including backward Euler and Crank–Nicholson, again without any restriction on the time step Δt [11].

As in section 2 for explicit methods, here we would like to discuss the possibility of designing higher order implicit methods that share the strong stability properties of backward Euler, without any restriction on the time step Δt .

Unfortunately, we are not as lucky in the implicit case. Let us look at a simple example of second-order implicit Runge–Kutta methods:

$$(6.3) \quad \begin{aligned} u^{(1)} &= u^n + \beta_1 \Delta t L(u^{(1)}), \\ u^{n+1} &= \alpha_{2,0} u^n + \alpha_{2,1} u^{(1)} + \beta_2 \Delta t L(u^{n+1}). \end{aligned}$$

Notice that we have only a single implicit L term for each stage and no explicit L terms, in order to avoid time-step restrictions necessitated by the strong stability of explicit schemes. However, since the explicit $L(u^{(1)})$ term is contained indirectly in the second stage through the $u^{(1)}$ term, we do not lose generality in writing the schemes as the form in (6.3) except for the absence of the $L(u^n)$ terms in both stages.

To simplify our example we assume L is linear. Second-order accuracy requires the coefficients in (6.3) to satisfy

$$(6.4) \quad \alpha_{2,1} = \frac{1}{2\beta_1(1-\beta_1)}, \quad \alpha_{2,0} = 1 - \alpha_{2,1}, \quad \beta_2 = \frac{1-2\beta_1}{2(1-\beta_1)}.$$

To obtain an SSP scheme from (6.4) we would require $\alpha_{2,0}$ and $\alpha_{2,1}$ to be nonnegative. We can clearly see that this is impossible, as $\alpha_{2,1}$ is in the range $[4, +\infty)$ or $(-\infty, 0)$.

We will use the following simple numerical example to demonstrate that a non-SSP implicit method may destroy the nonoscillatory property of the backward Euler method, despite the same underlying nonoscillatory spatial discretization. We solve the simple linear wave equation

$$(6.5) \quad u_t = u_x$$

with a step-function initial condition:

$$(6.6) \quad u(x, 0) = \begin{cases} 1 & \text{if } x \leq 0, \\ 0 & \text{if } x > 0, \end{cases}$$

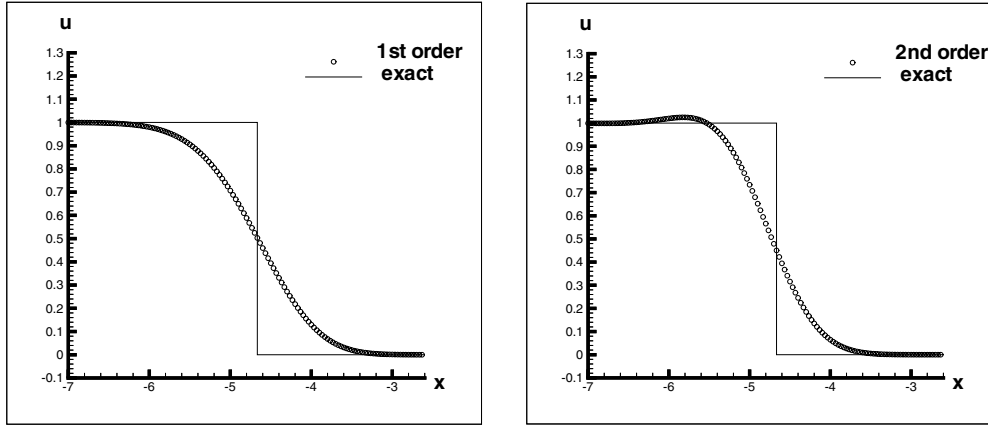


Fig. 6.1 *First-order upwind spatial discretization. Solution after 100 time steps at CFL number $\frac{\Delta t}{\Delta x} = 1.4$. Left: First-order backward Euler time discretization; right: non-SSP second-order implicit Runge–Kutta time discretization (6.3)–(6.4) with $\beta_1 = 2$.*

and u_x in (6.5) is approximated by the simple first-order upwind difference:

$$L(u)_j = \frac{1}{\Delta x} (u_{j+1} - u_j).$$

The backward Euler time discretization

$$u^{n+1} = u^n + \Delta t L(u^{n+1})$$

for this problem is unconditionally TVD according to Lemma 6.1. We can see on the left of Figure 6.1 that the solution is monotone. However, if we use (6.3)–(6.4) with $\beta_1 = 2$ (which results in positive $\beta_2 = \frac{3}{2}$, $\alpha_{2,0} = \frac{5}{4}$, but a negative $\alpha_{2,1} = -\frac{1}{4}$) as the time discretization, we can see on the right of Figure 6.1 that the solution is oscillatory.

In the next two subsections we discuss the rather disappointing negative results about the nonexistence of high-order SSP Runge–Kutta or multistep methods.

6.2. Implicit Runge–Kutta Methods. A general implicit Runge–Kutta method for (2.1) can be written in the form

$$\begin{aligned}
 (6.7) \quad & u^{(0)} = u^n, \\
 & u^{(i)} = \sum_{k=0}^{i-1} \alpha_{i,k} u^{(k)} + \Delta t \beta_i L(u^{(i)}), \quad \alpha_{i,k} \geq 0, \quad i = 1, \dots, m, \\
 & u^{n+1} = u^{(m)}.
 \end{aligned}$$

Notice that we have only a single implicit L term for each stage and no explicit L terms. This is to avoid time-step restrictions for strong stability properties of explicit schemes. However, since explicit L terms are contained indirectly beginning at the second stage from u of the previous stages, we do not lose generality in writing the schemes as the form in (6.7) except for the absence of the $L(u^{(0)})$ terms in all stages.

If these $L(u^{(0)})$ terms are included, we would be able to obtain SSP Runge–Kutta methods under restrictions on Δt similar to explicit methods.

Clearly, if we assume that the first-order implicit Euler discretization

$$(6.8) \quad u^{n+1} = u^n + \Delta t L(u^{n+1})$$

is unconditionally strongly stable, $\|u^{n+1}\| \leq \|u^n\|$, then (6.7) would be unconditionally strongly stable under the same norm provided $\beta_i > 0$ for all i . If β_i becomes negative, (6.7) would still be unconditionally strongly stable under the same norm if $\beta_i L$ is replaced by $\beta_i \tilde{L}$ whenever the coefficient $\beta_i < 0$, with \tilde{L} approximates the same spatial derivative(s) as L , but is unconditionally strongly stable for first-order implicit Euler, backward in time:

$$(6.9) \quad u^{n+1} = u^n - \Delta t \tilde{L}(u^{n+1}).$$

As before, this can again be achieved for hyperbolic conservation laws by solving (2.12), the negative-in-time version of (2.2). Numerically, the only difference is the change of upwind direction.

Unfortunately, we have the following negative result which completely rules out the existence of SSP implicit Runge–Kutta schemes (6.7) of order higher than 1.

PROPOSITION 6.2. *If (6.7) is at least second-order accurate, then $\alpha_{i,k}$ cannot be all nonnegative.*

Proof. We prove that the statement holds even if L is linear. In this case second-order accuracy implies

$$(6.10) \quad \sum_{k=0}^{i-1} \alpha_{i,k} = 1, \quad X_m = 1, \quad Y_m = \frac{1}{2},$$

where X_m and Y_m can be recursively defined as

$$(6.11) \quad X_1 = \beta_1, \quad Y_1 = \beta_1^2, \quad X_m = \beta_m + \sum_{i=1}^{m-1} \alpha_{m,i} X_i, \quad Y_m = \beta_m X_m + \sum_{i=1}^{m-1} \alpha_{m,i} Y_i.$$

We now show that, if $\alpha_{i,k} \geq 0$ for all i and k , then

$$(6.12) \quad X_m - Y_m < \frac{1}{2},$$

which is clearly a contradiction to (6.10). In fact, we use induction on m to prove

$$(6.13) \quad (1-a)X_m - Y_m \leq c_m(1-a)^2 \quad \text{for any real number } a,$$

where

$$(6.14) \quad c_1 = \frac{1}{4}, \quad c_{i+1} = \frac{1}{4(1-c_i)}.$$

It is easy to show that (6.14) implies

$$(6.15) \quad \frac{1}{4} = c_1 < c_2 < \cdots < c_m < \frac{1}{2}.$$

We start with the case $m = 1$. Clearly,

$$(1 - a)X_1 - Y_1 = (1 - a)\beta_1 - \beta_1^2 \leq \frac{1}{4}(1 - a)^2 = c_1(1 - a)^2$$

for any a . Now assume that (6.13)–(6.14), and hence also (6.15), is valid for all $m < k$, and that for $m = k$ we have

$$\begin{aligned} (1 - a)X_k - Y_k &= (1 - a - \beta_k)\beta_k + \sum_{i=1}^{k-1} \alpha_{k,i} [(1 - a - \beta_k)X_i - Y_i] \\ &\leq (1 - a - \beta_k)\beta_k + c_{k-1}(1 - a - \beta_k)^2 \\ &\leq \frac{1}{4(1 - c_{k-1})}(1 - a)^2 \\ &= c_k(1 - a)^2, \end{aligned}$$

where in the first equality we used (6.11), in the second inequality we used (6.10) and the induction hypotheses (6.13) and (6.15), and the third inequality is a simple maximum of a quadratic function in β_k . This finishes the proof. \square

We remark that the proof of Proposition 6.2 can be simplified, using existing ODE results in [5], if all β_i 's are nonnegative or all β_i 's are nonpositive. However, the case containing both positive and negative β_i 's cannot be handled by existing ODE results, as L and \tilde{L} do not belong to the same ODE.

6.3. Implicit Multistep Methods. For our purpose, a general implicit multistep method for (2.1) can be written in the form

$$(6.16) \quad u^{n+1} = \sum_{i=1}^m \alpha_i u^{n+1-i} + \Delta t \beta_0 L(u^{n+1}), \quad \alpha_i \geq 0.$$

Notice that we have only a single implicit L term and no explicit L terms. This is to avoid time-step restrictions for norm properties of explicit schemes. If explicit L terms are included, we would be able to obtain SSP multistep methods under restrictions on Δt similar to explicit methods.

Clearly, if we assume that the first-order implicit Euler discretization (6.8) is unconditionally strongly stable under a certain norm, then (6.16) would be unconditionally strongly stable under the same norm provided that $\beta_0 > 0$. If β_0 is negative, (6.16) would still be unconditionally strongly stable under the same norm if L were replaced by \tilde{L} .

Unfortunately, we have the following negative result which completely rules out the existence of SSP implicit multistep schemes (6.16) of order higher than 1.

PROPOSITION 6.3. *If (6.16) is at least second-order accurate, then α_i cannot be all nonnegative.*

Proof. Second order accuracy implies

$$(6.17) \quad \sum_{i=1}^m \alpha_i = 1, \quad \sum_{i=1}^m i\alpha_i = \beta_0, \quad \sum_{i=1}^m i^2\alpha_i = 0.$$

The last equality in (6.17) implies that α_i cannot be all nonnegative. \square

7. Concluding Remarks. We have systematically studied SSP time discretization methods, which preserve stability, in any norm, of the forward Euler (for explicit methods) or the backward Euler (for implicit methods) first-order time discretizations. Runge–Kutta and multistep methods are both investigated. The methods listed here can be used for method of lines numerical schemes for PDEs, especially for hyperbolic problems.

REFERENCES

- [1] M. CARPENTER AND C. KENNEDY, *Fourth-Order 2N-Storage Runge-Kutta Schemes*, NASA TM 109112, NASA Langley Research Center, June 1994.
- [2] B. COCKBURN AND C.-W. SHU, *TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws II: General framework*, Math. Comp., 52 (1989), pp. 411–435.
- [3] B. COCKBURN, S. HOU, AND C.-W. SHU, *TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws IV: The multidimensional case*, Math. Comp., 54 (1990), pp. 545–581.
- [4] G. DAHLQUIST, *A special stability problem for linear multistep methods*, BIT, 3 (1963), pp. 27–43.
- [5] K. DEKKER AND J. G. VERWER, *Stability of Runge-Kutta methods for stiff nonlinear differential equations*, Elsevier Science Publishers, Amsterdam, 1984.
- [6] S. GOTTLIEB AND C.-W. SHU, *Total variation diminishing Runge-Kutta schemes*, Math. Comp., 67 (1998), pp. 73–85.
- [7] D. GOTTLIEB AND E. TADMOR, *The CFL condition for spectral approximations to hyperbolic initial-boundary value problems*, Math. Comp., 56 (1991), pp. 565–588.
- [8] A. HARTEN, *High resolution schemes for hyperbolic conservation laws*, J. Comput. Phys., 49 (1983), pp. 357–393.
- [9] A. KURGANOV AND E. TADMOR, *New High-Resolution Schemes for Nonlinear Conservation Laws and Related Convection-Diffusion Equations*, UCLA CAM Report No. 99-16, University of California, Los Angeles.
- [10] Z. JACKIEWICZ, R. RENAUT, AND A. FELDSTEIN, *Two-step Runge-Kutta methods*, SIAM J. Numer. Anal., 28 (1991), pp. 1165–1182.
- [11] G. JIANG AND C.-W. SHU, *On cell entropy inequality for discontinuous Galerkin methods*, Math. Comp., 62 (1994), pp. 531–538.
- [12] B. VAN LEER, *Towards the ultimate conservative difference scheme V. A second order sequel to Godunov’s method*, J. Comput. Phys., 32 (1979), pp. 101–136.
- [13] D. LEVY AND E. TADMOR, *From semidiscrete to fully discrete: Stability of Runge–Kutta schemes by the energy method*, SIAM Rev., 40 (1998), pp. 40–73.
- [14] M. NAKASHIMA, *Embedded pseudo-Runge–Kutta methods*, SIAM J. Numer. Anal., 28 (1991), pp. 1790–1802.
- [15] H. NESSYAHU AND E. TADMOR, *Non-oscillatory central differencing for hyperbolic conservation laws*, J. Comput. Phys., 87 (1990), pp. 408–463.
- [16] S. OSHER AND S. CHAKRAVARTHY, *High resolution schemes and the entropy condition*, SIAM J. Numer. Anal., 21 (1984), pp. 955–984.
- [17] S. OSHER AND E. TADMOR, *On the convergence of difference approximations to scalar conservation laws*, Math. Comp., 50 (1988), pp. 19–51.
- [18] C.-W. SHU, *TVB uniformly high-order schemes for conservation laws*, Math. Comp., 49 (1987), pp. 105–121.
- [19] C.-W. SHU, *Total-variation-diminishing time discretizations*, SIAM J. Sci. Statist. Comput., 9 (1988), pp. 1073–1084.
- [20] C.-W. SHU AND S. OSHER, *Efficient implementation of essentially non-oscillatory shock-capturing schemes*, J. Comput. Phys., 77 (1988), pp. 439–471.
- [21] P. K. SWEBY, *High resolution schemes using flux limiters for hyperbolic conservation laws*, SIAM J. Numer. Anal., 21 (1984), pp. 995–1011.
- [22] E. TADMOR, *Approximate solutions of nonlinear conservation laws*, in Advanced Numerical Approximation of Nonlinear Hyperbolic Equations, Lectures Notes from CIME Course, Cetraro, Italy, 1997, A. Quarteroni, ed., Lecture Notes in Math. 1697, Springer-Verlag, New York, 1998, pp. 1–150.
- [23] J. H. WILLIAMSON, *Low-storage Runge-Kutta schemes*, J. Comput. Phys., 35 (1980), pp. 48–56.